VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2020 Michael Bláha

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ANALÝZA ŠKODLIVÉHO SOFTWARU
ANALYSIS OF MALWARE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Michael Bláha**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Vlastimil Člupek, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Michael Bláha ***ID:*** 195824
Ročník: 3 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Analýza škodlivého softwaru

POKYNY PRO VYPRACOVÁNÍ:

V bakalářské práci vytvořte metodiku pro analýzu škodlivého softwaru a navrhněte analyzační prostředí, ve kterém bude možné bezpečně analyzovat jakýkoliv škodlivý software. Vytvořte seznam bezpečnostních rizik, jenž může způsobit škodlivý software a určete jejich závažnost. Proveďte analýzu několika typů vybraných škodlivých softwarů ve Vámi navrženém analyzačním prostředí s využitím Vámi navržené metodiky. Určete bezpečnostní rizika a jejich závažnost, jenž může způsobit analyzovaný software. Výsledky analýz přehledně zpracujte a doporučte ochrany proti zkoumaným škodlivým softwarům.

DOPORUČENÁ LITERATURA:

[1] MOSER, Andreas; KRUEGEL, Christopher; KIRDA, Engin. Limits of static analysis for malware detection. In: Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007). IEEE, 2007. p. 421-430.

[2] DINABURG, Artem, et al. Ether: malware analysis via hardware virtualization extensions. In: Proceedings of the 15th ACM conference on Computer and communications security. ACM, 2008. p. 51-62.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 8.6.2020

Vedoucí práce: Ing. Vlastimil Člupek, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

Abstrakt

Cílem mojí bakalářské práce je navrhnout bezpečné prostředí pro analýzu škodlivého softwaru. V teoretické části práce se věnuji základnímu dělení počítačových virů. Poté popisuji dva hlavní postupy při analyzování škodlivého softwaru, a to statickou a dynamickou analýzu. Popisuji, z jakého důvodu se používají a jaké nástroje spadají do těchto kategorií. Dále prezentuji svoji metodologii pro bezpečnou analýzu škodlivého softwaru. V praktické části práce se věnuji vytvoření analytického prostředí na platformách Windows 10 a Fedora. Používám jak grafické prostředí, tak i příkazový řádek k vytvoření virtuálních počítačů. Abych mohl analyzovat síťový provoz, vytvářím takzvaný „falešný internet“ s programem INetSim. V poslední části práce se věnuji ukázce analýzy vybraných druhů počítačových virů. Postupuji podle mnou popsané metodologie. Ke každé analýze píšu krátké shrnutí a výsledky. Na konci práce se zabývám možnou obranu před škodlivým softwarem.

Klíčová slova

škodlivý software, počítačový virus, analýza, ransomware, trojský kůň, statická analýza, dynamická analýza, virtuální prostředí,

Abstract

The aim of my bachelor thesis is to design a safe environment for the analysis of malicious software. In the theoretical part of the work, I deal with the basic division of computer viruses. Next, I describe two main procedures for malware analysis, namely static and dynamic analysis. I describe why they are used and what tools fall into these categories. I also present my methodology for secure malware analysis. In the practical part of the work, I focus on creating an analytical environment on Windows 10 and Fedora platforms. I use a graphical environment and a command line to create virtual machines. For the analysis of network traffic, I create the so-called "Fake Internet" program with the INetSim program. In the last part of the work, I deal with a sample analysis of selected types of computer viruses. I follow the described methodology. For each analysis, I describe a brief summary and results. At the end of the work, I describe a possible defense against malicious software.

Keywords

malicious software, computer virus, analysis, ransomware, trojan, static analysis, dynamic analysis, virtual environment

Bibliografické citace

Internet Users' Glossary [online]. Austin, 1993 [cit. 2020-06-08]. Dostupné z: https://tools.ietf.org/html/rfc1392

JIROVSKÝ, VÁCLAV, 2007, *Kybernetická kriminalita: nejen o hackingu, crackingu,virech a trojských koních bez tajemství.*. Praha : Grada.

CHEBYSHEV, VICTOR and SINITSYN, FEDOR, 2019, IT threat evolution Q1 2019. Statistics. *Securelist.com* [online]. 2019. [Accessed 21 December 2019]. Available from: https://securelist.com/it-threat-evolution-q1-2019-statistics/90916/

KUJAWA, ADAM and ZAMORA, WENDY, 2019, [online]. Malwarebytes. [Accessed 21 December 2019]. Available from:

https://resources.malwarebytes.com/files/2019/01/Malwarebytes-Labs-2019-State-of-Malware-Report-2.pdf

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Analýza škodlivého softwaru“ jsem vypracoval samostatně pod vedením vedoucího semestrální práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené semestrální práce dále prohlašuji, že v souvislosti s vytvořením této semestrální práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení *S* 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Vlastimilu Člupkovi, Ph.D. za velmi užitečnou metodickou pomoc a cenné rady při zpracování práce.

Brno ....
podpis autora

Obsah

Úvod

Se slovy hacker, virus se dneska potkáme na každém rohu, jsou nedílnou součástí našich každodenních životů a můžou mít dopad i na náš budoucí život. Ač si mnoho lidí neuvědomuje závažnost kyberkriminality a škody napáchané ilegální činností, přesto si myslím, že hlavně díky mladším generacím se povědomí o této hrozbě zvyšuje. Hackerem se dnes může stát prakticky každý, jelikož jsou dostupné kurzy, které vám umožní vzdělání v pronikání do špatně zabezpečených zařízení i přesto, že nejsou zaměřené na lidi, co chtějí porušovat zákon, ale naopak pomáhat lidem a firmám k obraně proti nim.

V první polovině práce, bych rád čtenáře seznámil se základní terminologií, představil mu některé skupiny škodlivého kódu. Bude se jednat konkrétně o spyware, adware a trojského koně. Důvodem, proč jsem si vybral právě tuto skupinu je jeho ohromné rozšíření po světě, mnohdy nepozorované provádění nechtěné činnosti na počítači oběti, jako odcizení citlivých dokumentů, zaznamenávání hesel a zobrazování nechtěných reklam. Také díky schopnosti umožnění šířit další malware buď na tento počítač nebo dále po síti.

Mým cílem je ukázal přípravu bezpečného prostředí pro jeho analýzu. Jak pro platformu Windows 10, tak i pro distribuci linuxu Fedoru. Budu se hlavně věnovat práci s příkazy, v rámci powershellu a bashe. Dále ukážu některé analytické programy, které budu později využívat.

Pro lidi, co by si s malwarem chtěli hrát a zlepšovat se v jeho analýze a porozumění tomu, jak funguje a co mohlo vést tvůrce k vytvoření právě takové malwaru. popíšu také, co by si začátečník v tomto oboru měl uvědomit a připravit, než si stáhne náhodný virus z internetu a infikuje si počítač.

Dále bych poté poukázal na malware, který představuje aktuální nebezpečí. Doložil bych svoje tvrzení na datech ze světa ve formě grafů a tabulek.

V závěrečné části práce předvedu mnou navržené bezpečné prostředí. Ukážu, jak pracovat s programy, které jsem na začátku představil. Uvedu možnou obranu proti analyzovaným hrozbám.

1 Základní pojmy

Škodlivý software (malware) je takový program, který má za úkol poškodit nebo narušit virtuální prostor oběti. Dokáže se šíři a napadat systémy v počítači nebo jiné počítače nepozorovaně, bez vědomí majitele. V dnešní době je používání škodlivého softwaru velice populární, a můžeme ho najít prakticky v každém elektronickém zařízení co má alespoň malou výpočetní kapacitu například: všechny počítače, chytré telefony, ale i chytré spotřebiče, zdravotní zařízení, routery. Díky jednoduchému přístupu k informacím, pomocí internetového prostředí, můžeme najít škodlivý software během několika okamžiků. Stačí pouze vědět co hledat. Tím pádem používat škodlivý software dokáže dneska už i malé dítě, jelikož si stačí stáhnout už předem vytvořený kód a jenom ho použít. Dalším velikým problém je, že mnoho lidí nevěnuje mnoho času a prostředků zabezpečení svých zařízení. Toto je hlavně problém u firem, které často používají zastaralé již nepodporované operační systémy nebo programy, protože jsou zvyklí pracovat s tím jedním prostředkem, popřípadě nechtějí investovat prostředky na aktualizaci [1].

1.1 Hacker vs cracker

Jako první si musíme rozdělit a ujasnit rozdíly mezi tím co znamená hacker a cracker, jelikož se často tyto dva pojmy zaměňují. Pokud dneska člověk řekne slovo hacker, tak si každý představí programátora, co se snaží působit škodu, převážně prostřednictvím internetu, aby se obohatil na úkor druhých[2].

RFC 1392: Internet Users' Glossary definuje slovo "hacker" jako:

„A person who delights in having an intimate understanding of the internal workings of a system, computers and computer networks in particular“ (Jirovský 2007).

Což volně přeloženo znamená: „Osoba, která si libuje v prozkoumávání systémů a jeho funkcí do úplného detailu, zejména počítačů a počítačových sítí.“ Už od prvního pohledu na tuto citaci si můžeme všimnout, že zde není řečeno nic o působení nějaké škody nebo nelegální pronikání do cizích počítačových systémů.

RFC 1392: Internet Users' Glossary definuje slovo "cracker" jako:

„A cracker is an individual who attempts to access computer systems without authorization. These individuals are often malicious, as opposed to hackers, and have many means at their disposal for breaking into a system“ (Jirovský 2007).

Pokud bych i tuto definici měl volně přeložit: „Cracker je osoba co se snaží získat neoprávněný přístup do počítačového systému.“ Tyto jedinci jsou často nebezpeční na rozdíl od hackerů a mají k dispozici mnoho prostředků pro proniknutí do systému. Abych shrnul tyto dvě definice, hacker nemusí nutně být člověk snažící se nelegálně pronikat do počítačových systémů. V dnešní době hackery rozlišujeme do tří skupin.

První skupinou jsou takzvaní „black hat hackers“, o nich můžeme mluvit jako o „crackerech“ právě pro to, že jejich hlavní motivací je finanční zisk nebo osobní prospěch[3].

Na druhou stranu „white hat hackres“, zmámí také jako „ethical hackers,“ jsou to lidé, co pracují buď jako zaměstnanci pro firmy nebo pracují jako kontraktoři, kteří se snaží cíleně napadat počítačové sítě jejich zákazníků ve snaze najít a nahlásit nedostatky v zabezpečení počítačových sítí. Používají stejné prostředky jako black hat hackers, dělají to však s vědomím zákazníka, tudíž se nejedná o nelegální činnost. Ohledně vzdělávání v této oblasti je možné chodit na kurzy nebo pokud v tom jsou opravdu dobří, můžou se pokusit udělat certifikáty, jako například: CEH (Certified ethnical hacker) [3].

Do třetice tu máme hackery, kteří hledají slabiny v zabezpečení, ale dělají to bez povolení a s vidinou zisku, ale na rozdíl od „black hat hackerů“ dávají možnost firmám za úplatu na dané problémy ukázat, popřípadě je rovnou vyřešit. Na druhou stranu, pokud nedostanou to, co chtějí, tak neváhají dané nedostatky v zabezpečení prodat za nejvyšší nabídku na „dark webu“ nebo použít k vlastním cílům. Tato skupina lidí zvaní „grey hat hackers“ žije na hraně zákona [2][3].

1.2 Prostředí pro analýzu škodlivého kódu

Jednou z prvních věcí, které by se člověk, věnující se analýze škodlivého kódu, měl věnovat je virtuální prostředí. Jedním typem, který se využívá nejvíce pro analýzu je takzvaný sandbox. Jelikož je to nezbytné z hlediska bezpečnosti

a omezení šíření malwaru. K vytvoření virtuálního prostředí můžeme použít jeden z mnoha nástrojů například: KVM, Virt, VM-Vare, Virtual Box. Virtualizaci lze provádět na různých úrovní, od virtuálního stroje, přes jednotlivé komponenty po software. Virtualizace umožní přistupovat ke zdrojům (hardware), jiným způsobem, než je fyzické propojení. Tím pádem, ho můžeme upravit, tak aby programy běžící uvnitř nemohly zasahovat do hostitelského počítače. V dnešní době můžeme používat jakýkoliv operační systém například: Windows, distribuce linuxů (Fedora, Debian), MacOs, ale i mobilní operační systémy jako: IOS, Android. Dále v práci se budu věnovat více dopodrobna některým vybraným.

2 Škodlivý software

V této kapitole práce, přiblížím pár základních typů škodlivého kódu, jako jsou: Počítačový červy, Adware, Spyware, Ransomware, Trojský kůň, Rootkyt. Vysvětlím jejich základní funkci, uvedu jejich příklad.

2.1 Počítačový červy

Základní charakteristikou počítačového červa je jeho sebe-replikační schopnost. Snaží se rozšiřovat a napadat zařízení přes počítačovou síť. První červy byly vytvořeny již v 70. letech 20. století. Mezi lety 1979-1981 výzkumníci z Xerox PARC experimentovaly s červy, za účelem šíření se mezi jednotlivými zařízeními a provádění užitečných operací. První červy nebyly nazvány viry, protože lidé se domývali, že nejsou vytvořené za účelem škodit. Jenže záměry, se kterým červy jsou vytvořené, a efekt jaký mají na systém, jsou značně rozličné, tudíž i červ může spadat pod označení virus. Ve státě Liusiane v USA se stala objetí internetová společnost WebTV, která byla napadena počítačovým červem od neznámého útočníka. WebTV poskytoval internet skrze televizní spojení. Po infikování červem, zařízení u zákazníků přednastavilo číslo k internetovému připojení na 911´, což je linka nouzového volání. Kvůli tomuto napadení bylo způsobeno veliké množství planých poplachů [4].

2.2 Adware

Software, jehož jediným účelem je zobrazovat nechtěné reklamy na napadeném počítači. Je zde více způsobů, jakým adware zobrazuje reklamy, například: vyskakující panely v prohlížeči, otevírající se okna prohlížeče s reklamou, změní domovskou stránku prohlížeče za reklamní stránku. Jak si můžeme všimnout v obrázku 1, okna s reklamami se zde zobrazují skoro na celé ploše monitoru a ruší tak zážitek z prohlížení internetu. Adware se nejčastěji šíří s volně stažitelným softwarem, klamavými stahovacími klienty a instalátory. První adwary začaly působit už v 90. letech 20 století, kdy byly ještě považovány za spyware. Největší rozšíření zažil adware mezi lety 2005-2008, kdy začal být monitorován a zvýšila se snaha omezit ilegální činnost [5].

Obr. 2.1: Napadený počítač adwarem zobrazuje v prohlížeči Internet Explorer nechtěná reklamní okna.

2.3 Spyware

Slovo spyware poprvé použila společnost Zone Labs v roce 1999, kdy byl význam spojen s počítačovým softwarem. Do té doby byl spojován s monitorovací a špionážní technikou. Steve Gibson, vývojář ve společnosti OptOut, popsal spyware jako “Any software that employs a user’s Internet connection in the background (the so-called ‘backchannel’) without their knowledge or explicit permission.”

Hlavním účelem spywaru je sledovat, zaznamenávat a odesílat celkové dění na napadeném zařízení. Dělí se na 4 typy: adware, trojské koně, software pro zaznamenávaní cookies a monitorování zařízení. Pro spyware je nejdůležitější, aby zůstal nedetekovaný a mohl tak vykonávat svoji činnost. Většinou nemá sebe replikační schopnost, takže se nešíří sám na rozdíl od viru a červů. Místo toho se šíří buď s volně stažitelným softwarem nebo skrze využívání slabin v zabezpečení softwaru. Nejběžnějším typem, jsou takzvané „keylogery,“ které zaznamenávají především údery na klávesnici [5].

2.4 Ransomvare

Termín vznikl spojením dvou anglických slov ransom a software, což doslovně přeloženo znamená „program požadující výkupné.“ Tento překlad je však hodně nepraktický, a proto se využívá anglického slova. První útok definovatelný jako napadení počítačů ransomwarem se odehrál v roce 1989 a zasáhl skoro všechny vyspělé země. Přenašečem počítačového viru se staly pěti-palcové diskety, které byly rozesílané z Londýna. Poštovní adresy byly podvodně vylákány z vydavatelů

technicky zaměřených časopisů. Disketa obsahovala dotazník s otázkami zaměřenými na nemoc AIDS. Po spuštění dotazníku ransomware zašifroval všechny názvy souborů a tím vyřadil napadený počítač. Pro získání klíče oběti museli zaplatit 189 dolarů a zaslat je na účet vedený v Panamě.

Ransomware nemá schopnost sám se rozšiřovat. Pro napadení počítače musí využít jiného počítačového viru nebo musí spoléhat na to, že uživatel si ho sám stáhne. Třeba prostřednictví emailu nebo infikovaného programu. Po úspěšném infikování cílového počítače, ransomware může začít pracovat okamžitě, nebo může čekat na povel ke spuštění. Útok není okamžitý, zašifrování dat, obzvláště pokud se jedná o větší množství dat, trvá nějakou dobu. Po zašifrování dat se na monitoru zobrazí upozornění 0 napadení a je žádáno výkupné. Typ zobrazované správy záleží na druhu ransomwaru, může to být všechno od domnělé pokuty přes poplatek za odstranění počítačového viru až po žádost o výkupné. Výkupné je placeno na zahraniční účet, ale v posledních letech se rozšířila platba prostřednictvím kryptoměn, díky jejich obtížnému nalezení vlastníka. Po zaplacení útočníci zašlou heslo k dešifrování souborů, a tak možnému opětovnému použití počítače bez ztráty uživatelských dat[6].

2.5 Trojský kůň

Útok pojmenovaný podle legendární zrady Řeků na Trojanech, který se snaží vypadat jako užitečný software. Převážně nejde o komplikovaný vir, nemá schopnost sebe-replikace. Jedním z prvních virů spadajících do této kategorie se jmenuje EGABTR. Největší rozsah měl na přelomu let 1980–1990. Jeho cílem bylo mazat soubory FAT (file allocation tables) na pevném disku. Po smazání dat se uživateli zobrazil nápis „Arf! Arf! Got you!" Za zmínku stojí 4 druhy [7]:

2.5.1 Downloader

Funkce, jak už napovídá jeho jméno, je stahovat další škodlivý software do zasaženého počítače. Většinou se jedná o složitější formy škodlivého softwaru, jako například: adware nebo spyware. Je šířen převážně pomocí e-mailu, díky jeho jednoduchosti a nulové nebezpečnosti kódu anti-malwarové programy mají problém s jeho detekcí a likvidací [8].

2.5.2 Backdoor

Umožňuje cílovým softwarům vyhnout se zabezpečení na počítači. Nejčastěji jde o umožnění vzdáleného přístupu a poté získání kontroly nad daným zařízením. Ovládnutému zařízení se říká „zombie.“ V opačném případě lze programy počítané za trojské koně použít ke vzdálenému přístupu a administraci. Jeno příkladem může být Remote Access Trojan zkráceně RAT, slouží k monitorování a ovládání vzdáleného zařízení.

2.5.3 Password-stealing trojan

Hlavním cílem je snaha odcizit hesla, a to buď přímo ze souborů na napadeném počítači nebo využitím keyloggeru. Získaná data jsou následně odeslána buď formou e-mailu na útočníkovu adresu nebo odeslána skrze IRC (Internet Relay Chat) [8].

2.5.4 Proxy trojan

Umožňuje napadený počítač proměnit v podstatě v proxy server. Nejčastěji je používán k rozšiřování spamových a podvodných e-mailů při zachování útočníka skrytého.

2.6 Rootkyt

Vyvinul se z jednoduchého souboru nástrojů známí jako „Rootaccess“, který umožňoval přístup k zařízení UNIX na úrovní administrátora. Obsahoval nástroje jako: „ps, netstat, ls, and passwd,“ které se běžně používají pro správu operačního systému a přístupu k síti. Jenže tyto nástroje začali používat hackeři k maskování stop, které zanechali v napadeném systému. Po čase se napadení nevyhnul ani operační systém Windows, název však zůstal stejný rootkyt. Můžeme je dělit do kategorií podle vrstvy v zařízení na které fungují [9].

2.6.1 Uživatelský rootkit

Snaží se napadat nejnižší stupeň oprávnění operačních systémů. Snaží se měnit běžící programy za pomoci hákování („hooking“), tedy snaží se zaměnit

výstupy volání funkcí uživatele. Útok se odehrává v paměti počítače, se kterými pracuje procesor. Napadány jsou soubory s příponami .exe, .obj a .dll.

2.6.2 Rootkity modulu jádra

Fungují jako modul operačního systému a využívají tak nejvyšší oprávnění. Napadení jádra je velice lukrativní pro útočníky, jelikož jsou schopni zasahovat do celého dění v daném systému. Spadají zde 3 základní typy útoku: Hákování System Service Table (SST hooking), Manipulace tabulky popisující přerušení (Interupt Description Table manipulation), Přímá manipulace s objekty jádra (Direct Kernel Object manipulation) [10].

2.6.3 Hardwarový rootkit

Virus se ukrývá a napadá samotné komponenty v počítači, respektive jejich firmware. Tímto útokem lze napadnou třeba BIOS počítače nebo měnit mikro kód počítače. Detekce je velice náročná někdy skoro nemožná. Jelikož dochází k napadání firmwaru, který běží na nižší úrovni než operační systém, dále výrobci jednotlivých komponent nezveřejňují nástroje nebo zdrojové kódy. Samotné programování takové viru vyžaduje značnou míru zkušeností a zdrojů.

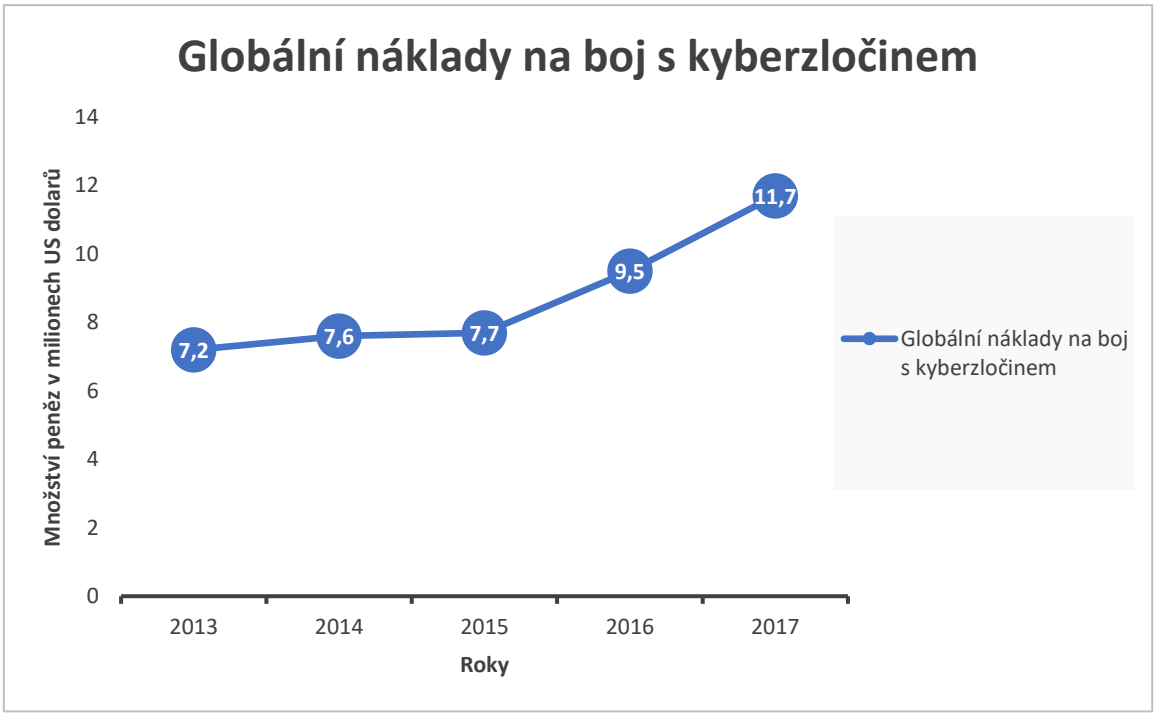
2.6.4 Rootkity řízeného kódu

Nejnovější skupina rootkitů, zatím se v praxi moc nevyskytuje. Pracuje na úrovni uživatelského právnění. Cílem je napadat virtualizované prostředí nebo běhové prostředí. Týká se to programovacích jazyků, třeba C#, Java nebo Python. Rootkit napadá knihovny jednotlivých jazyků buď jako zdrojový kód nebo předkompilovaný bytecode. Díky špatné mnohdy neexistující optimalizaci bytecode, tak se je možné jej převést zpět do zdrojového kódu. Díky tomu útočník může zasahovat do knihoven a měnit jejich funkčnost. Neexistují nástroje na odhalení, většinou jsou cílem vysoce výkonné počítače. Úroveň virtualizace je zde mnohem komplexnější a rozšířenější než u běžných uživatelů [10].

3 Analýza škodlivého softwaru

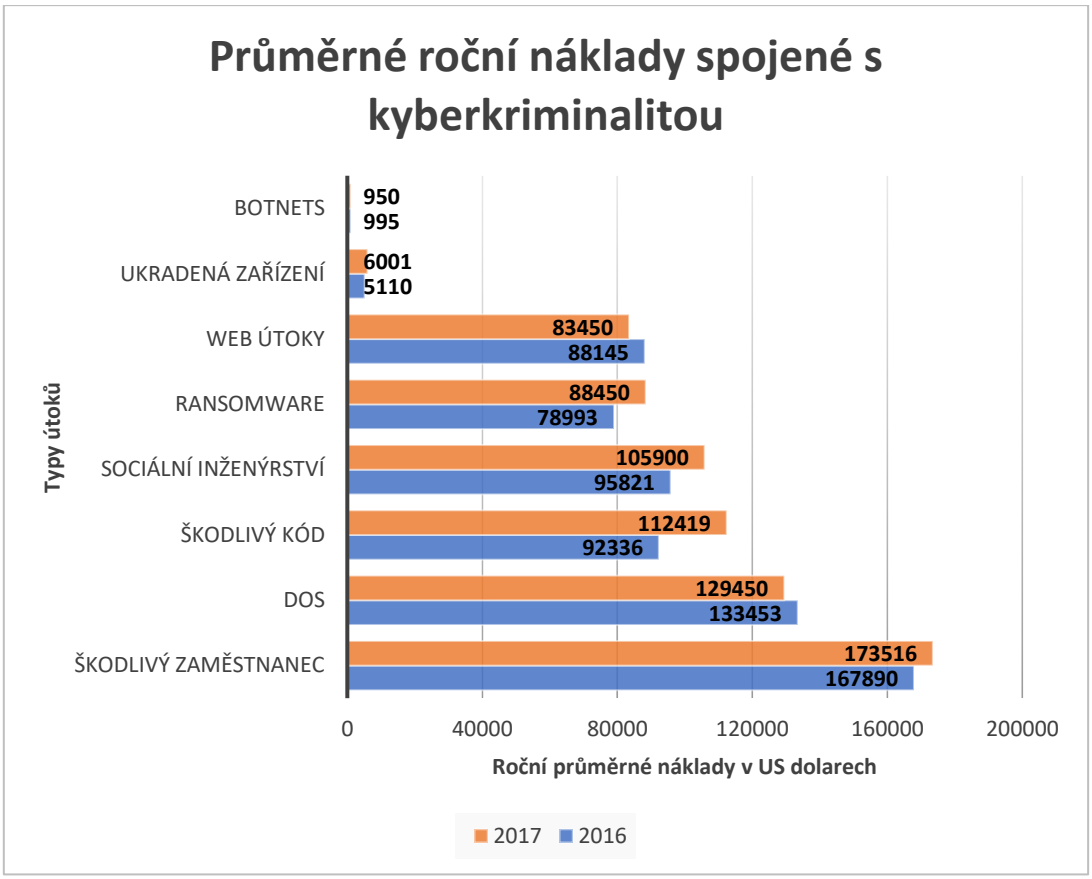
Jak už název napovídá, v této kapitole si povíme něco analýze škodlivého kódu, proč se jí věnujeme a jaké máme druhy. Analýza je proces, při kterém získáváme znalosti o určité věci. Důvod, proč analyzujeme škodlivý kód je abychom získali informace o jeho fungování, popřípadě se snažíme odhalit jeho záměr. Při analýze se snažíme škodlivý kód zkoumat co nejvíce dopodrobna, zjistit co jednotlivé komponenty dělají, jestli posílá nějaká data přes síť. Po provedení analýzy můžeme navrhnou vhodná protiopatření, tak aby už k napadení nemohlo dojít. Popřípadě navrhneme postup při likvidaci malwaru z napadeného počítače.

V dnešní době je kladen veliký důraz na bezpečnost virtuálních dat, zároveň se zvyšují rozpočty firem na boj s kyberzločinem. Jak tento graf ze studie (Obr. 3.1): „COST OF CYBER CRIME STUDY 2017“ [11] ukazuje, tak mezi roky 2013 a 2017 došlo k nárůstu peněz, které byly investovány do boje s kyberzločinem vzrostl o 62 %.



Obr. 3.1: Graf ukazující nárůst peněz vynaložených na boj s kyberzločinem mezi roky 2013-2017

Ještě bych tu rád ukázal jeden graf (Obr. 3.2), a to je průměrná roční cena, která připadá na jednotlivé typy útoků vůči jejich počtu [11].



Obr. 3.2: Průměrné roční náklady spojené s kyberútoky

V knize Practical Malware Analysis [12], autoři popisují tři pravidla, jak postupovat při analýze škodlivého kódu:

1. Snažit se najít klíčové funkce a podstatné vlastnosti. Nevěnovat se přílišným detailům, jelikož pokud by se jednalo o komplexnější a rozsáhlejší malware, tak by nám to zabralo příliš mnoho času. Udělat si celkový obrázek o chování a cíli malwaru.
2. Nepoužít furt jen jeden postup a jeden nástroj. V dnešní době je přístupných nesmírné množství nástrojů, mnoho návodů a postupů, jak při analýze škodlivého kódu postupovat. Když nám nejde dosáhnou úspěchu za použití jednoho nástroje a nějaké metodiky, tak i když jiný nástroj má podobné funkce tak může přinést průlom. Zároveň pokud se zasekneme, tak než se snažit řešit problém stále stejným způsob, je dobré se na věc podívat s odstupem a zkusit zvážit jiný postup.
3. Analyzování škodlivého softwaru není jednoduchá práce. S novými způsoby obrany a analýzy se zároveň komplikují i malwary a naopak. Je to vlastně taková válka ve zbrojení. Hackeři jsou vždycky o krok napřed,

jelikož dokud oni nevytvoří malware tak většinou není způsob, jak se bránit.

3.1 Statická analýza

Statická analýza je ideálním základem pro získávání informací o malwaru. Využívá metody, které nevyžadují spuštění malwaru. Díky tomu, že nemusíme spouštět malware a vyhneme se tak nechtěnému přenosu malwaru na počítač nebo jeho úniku ze sandboxu, tak je považována za bezpečnější než dynamická analýza.

Proces statické analýzy má mnoho podob a způsobů jakými získávat informace od analyzování metadat až po zkoumání každé instrukce a rozluštění významu jednotlivých bloků programu. Malware prakticky ve všech případech je compilován, to převedení zdrojového kódu do binárního, například přípona .exe. Získávání informací z kompilovaného kódu je mnohem náročnější než ze zdrojového kódu.

3.1.1 Fingerprinting

Zabývá se problémem rozlišováním jednotlivých vzorků malwaru. Jelikož porovnávání podle jmen souborů je z hlediska množství typů nevhodné. K identifikaci se tedy používají řetězce znaků zvané také jako otisky nebo hashe. Hash je jednosměrná matematická funkce, která z jakkoliv dlouhých vstupních dat udělá stejně dlouhý výstup. Hashe se ukládají do tabulek nebo databází kde jsou k dispozici k porovnání. Nejčastěji se používají kryptografické hashovací funkce například MD5 (The Message-Digest Algorithm), SHA-1(Secure Hash Algorithm), SHA-256. Hashe MD5 a SHA-1 se jsou považovány za prolomené, a nebylo by vhodné nimi šifrovat žádná důležitá data [12].

V dnešní době jsou existují rozsáhlé databáze obsahují prakticky všechny známé malwary. Většinou jsou tyto databáze vedeny společnostmi, které se věnují zabezpečení elektronických zařízení jako: AVAST, ESET a mnoho dalších. Fingerprinting se také používá při dynamické analýze, kdy spuštěný malware mohl stáhnou další malware nebo se naklonoval. Kde umožní ověření, jestli jde o původní malware nebo jestli byl modifikován.

Hlavní výhodou tradičních kryptografických hashovacích funkcí je náchylnost na změnu hashe, tedy pokud dojde jenom k malé úpravě malwaru tak se hash změní. Což není dobrý, pokud se snažíme uspořádat skupinu podobných malwarů, tak aby měli podobné hashe. S tímto problém nám mohou pomoct takzvané fuzzy hashe příklad: ssdeep, CTPH (není přesně fuzzy hash, ale algoritmus, ze které fuzzy has vychází). Hlavní rozdíl je ve vytvářením hashů, kdy fuzzy algoritmus generuje pro podobné vstupní řetězce podobné hashe. Díky tomu je můžeme porovnávat navzájem si podobný malware.

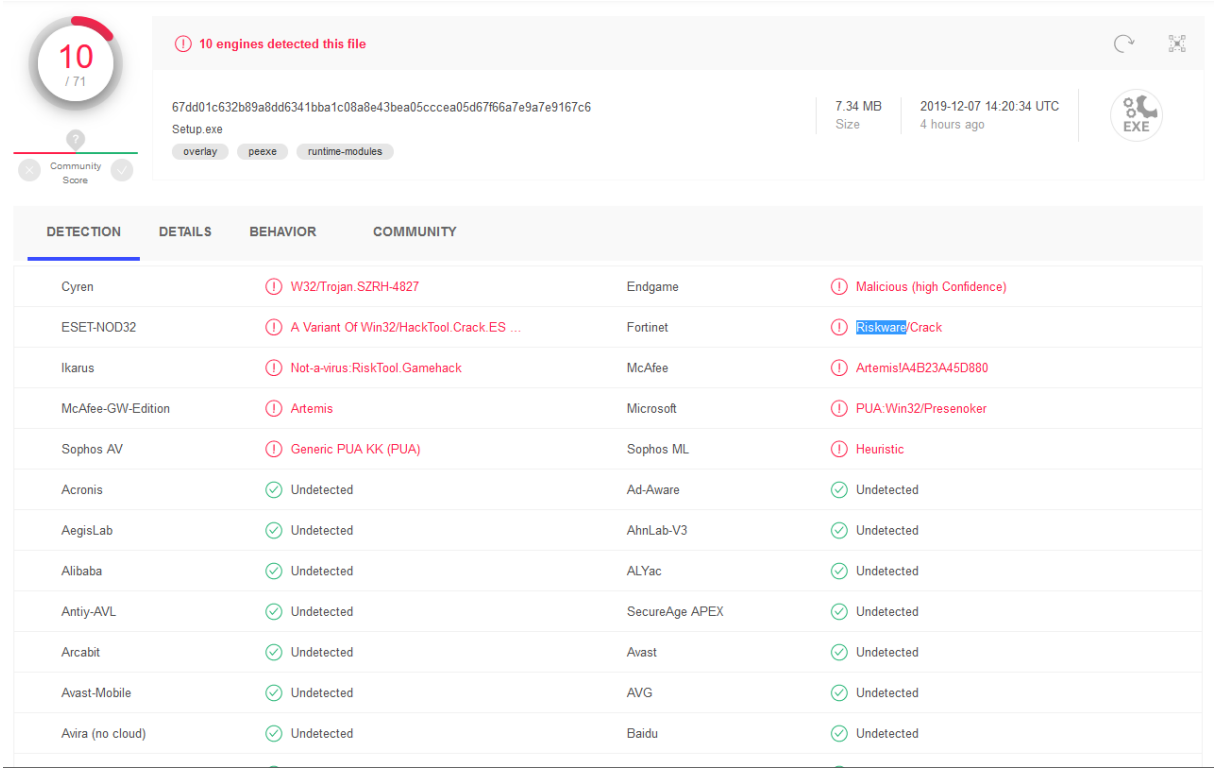
Další skupinou užitečných hashů během analýzy malwaru je takzvaný imphash (import hash) [13]. Používá se k podepisování souborů PE (Portable Excutable), což je souborový formát pro spustitelné soubory .exe, dynamické knihovny .dll, a objektové soubory .obj. Imphash je výpis MD5 hashe ze všech knihoven, které PE soubor importuje. V mnoha případech byl používán k přesnému spojení souboru PE viděného v jednom prostředí na soubory PE v jiných prostředích i přestože obsah každého z těchto souborů PE byl jiný. Důvod proč se imphash používá je, že změna iphashe odvozeného artefaktu v PE souboru je nepravděpodobná, jelikož ke změnění artefaktů by se musel změnit zdrojový kód, překompilovat. Což je nákladný a zdlouhavý proces. Authentihash se používá k ověření autentičnosti daných částí PE souborů. Vypočítává se z PE hlavičky za pomoci SHA-256. Položky ve hlavičce jsou seřazeny v daném pořadí a vynechává se kontrolní součet.

3.1.2 Antivirové skenery

Před začátkem nějaké hlubší analýzy je dobré prvně nechat zkontrolovat daný malware již existujícím antivirovým skenerem [12]. Skener se pokusí porovnat hash malwaru s hashem v jeho databázi a následně zobrazí informaci, zde se jedná o škodlivý software nebo ne. Všechny antivirové skenery nemusí obsahovat stejnou databázi, takže v případě neúspěšné detekce u jednoho, může za to sát vyzkoušení jiného. Pokud se bude jednat o úplně nový či výrazně modifikovaný malware, tak ani i při použití více skenerů nemusí dojít k detekci. Dneska už existují služby, které na daném vzorku provedou automatickou kontrolu u mnoha různých skenerů, takže se s tím nemusíme dělat ručně. Zase jako předtím, po dokončení kontroly jsou zobrazeny výsledky. Které obsahují

mnoho užitečných informací, jako jméno malwaru, které bylo přiřazeno jednotlivými antivirovými programy, jaký tip malwaru obsahuje, a jakou hrozbu představuje. Viz obrázek (Obr. 3.3), kde můžeme vidět výsledky ze souboru Setup.exe, ve kterém jsou pravděpodobně schované nějaké hrozby. Když uživatel tento soubor zpustí, tak se jeho zařízení infikuje.

Jednou z online služeb je nástroj jménem VirusTotal [14]. Zde člověk může nahrát soubor, nebo zadat URL adresu zkoumaného souboru. Následně VirusTotal udělá hash z daného souboru. Následně hash porovná s databází hashů, kterou získávají od jednotlivých antivirových firem(více než 70). Dalšími podobnými službami jsou AVCaesar, Jotti's malware scan, OPSWAT Metadefender Cloud. Všechny nabízejí podobnou funkčnost jediné, v čem se zásadně liší, je počet antivirových firem, které spolupracují s danou službou.



Obr. 3.3: Ukázka provedeného skenu na stránce www.virustotal.com.

Používání nástrojů třetích firem může mít i nějaká úskalí. Když nahrajeme nějaký soubor na jejich stránku, tak nemůžeme zaručit, jestli daná stránka nezneužije soubor pro sebe. U malwaru cíleného na určitou firmu, nebo druh zařízení, nemůžeme vyloučit, že v kódu nejsou citlivé údaje o firmě nebo o

programu například: IP adresy, hesla, přihlašovací údaje. Aby člověk nemusel nahrávat přímo daný soubor, některé nástroje umožňují nahrát jen hash.

3.2 Dynamická analýza

Dynamická analýza je založená na aktivním zkoumání běžícího malwaru v nějakém sandboxovém prostředí, které statická analýza nutně nevyžadovala. O sandboxu se pobavíme v další kapitole. Po spuštění malwaru, za pomoci specializovaných nástrojů pozorujeme jeho chování na následně z něho můžeme vyvodit závěry například: jak dále postupovat, jakou hrozbu daný malware představuje, nebo jakým způsobem byl pozměněno sandboxové prostředí [15].

Ohledně porovnávání změn, máme zde dvě možnosti. Zaprvé monitorovat změny v reálném čase, kdy sledujeme systémové volání (hook-based princip) nebo registrování oznámení (notification-basedprincip). Sledování změn nám, ale komplikuje běžná činnost systému a procesy vykonávané na pozadí nebo programy třetích stran. Pokud chceme tyto nechtěné změny odfiltrovat, tak je dobré si dopředu zjistit a pozorovat jaké tyto aplikace dělají změny. Potom můžeme bezpečně určit, které změny jsou způsobené malwarem a které obyčejnými programy [16].

3.2.1 Hook-based nástroje

Nástroje pracující na principu „hook," neboli uloží se do rozraní API buď v uživatelském nebo v režimu jádra. Funkce API jsou již na programované celky, skládající se z procedur, funkcí tříd, protokolů a knihoven. V grafice se třeba jedná o API OpenGL a DirectX. Následně potom zaznamenávají a ukazují změny probíhající v systému. Mezi jejich nevýhody spadá náročné programování a problémy s debugem. Příkladem těchto nástrojů je Process monitor, Hook analyzer a pymon.py.

3.2.2 Difference-based nástroje

Do češtiny je můžeme přeložit jako „instalační monitory." Jejich hlavní funkce pořízení takzvaného „snapshotu," neboli pořídí kopii nastavení systémů a registrů před a po spuštění nějakého programu. Poté porovnávají tyto dva záznamy

a vyhodnocují, jestli došlo ke změnám, popřípadě jakým. Jako příklad si můžeme uvést Regshot, Winanalysis a InCtrl5.

3.2.3 Notification-based nástroje

Hlavním cílem těchto nástrojů je zaznamenávání oznámení systému, které provádí automaticky pokud se uskuteční nějaká událost. Jako je vytváření souborů, přesouvání, změna práv uživatele, mazání záznamů v registrech. Hlavním představitelé jsou Process monitor a Preservation.

3.2.4 Zaznamenávání API volání s Process Monitor aplikací

Nástroj Process monitor vznik kombinací dvou jiných nástrojů, a to Filemon a Regmon. Používá se k zaznamenávání podrobných informací ovlivňující souborový systém, registry, síť a procesy. Jedná se o hybrid mezi hook-base a notification-base nástroji [17]. Funguje tak, že nahraje ovladač jádra, který se uloží („hookne“) funkce, například ZwDeleteKey nebo ZwSetValueKey, u kterých monitoruje registry. Dále používá ETW (Event Tracking for Windows) pro zaznamenávání síťové aktivity a aktivity vláken. Process Monitor má funkci „Boot login,“ díky které dokáže zaznamenávat API při zapínání počítače. To je obzvlášť užitečné, když se snažíme odhalit malware, který se zapíná při startování počítače. Ten dokáže provést nežádoucí změny ještě před tím, než jsou diagnostické nástroje vůbec zapnuté. Začíná zaznamenávat již při vytvoření smss.exe, což je první proces uživatele. Tím pádem zaznamenává věci, které se staly ještě přes spuštěním procesů jako crss.exe (Client/Server Run-Time Subsystem), winlogon.exe (Windows NT Logon Application) a explorer.exe (Windows Explorer). Process monitor má základně 7 sloupců obsahující data:

Time of day: Čas, kdy byla daná událost zaznamenaná. Může také zobrazovat čas od předchozí události.

Process name: Jméno procesu, vytvořené podle názvů daného procesu.

PID: ID (unikátní identifikátor) procesu.

Operation: Název spuštěné API, nebo krátký popis.

Path: Cesta k objektu, který byl spuštěn.

Result: Výsledek operace (úspěch/neúspěch).

Details: Detaily týkající se operace, jako úroveň přístupu.

3.2.5 Detekování změn s programem Regshot

Regshot je opensource, different-base nástroj, zaměřený na souborový systém a registry. Podobné programy jsou InCtrl5 a Winanalysis. Na rozdíl od zmíněných programů je Regshot mnohem rychlejší a nepotřebuje instalaci [18].

Po spuštění Regshot, dojde k pořízení prvního „snapshotu," který slouží jako základ k dalšímu porovnání. Jsou použity RegEnumValues a RegEnumKeyEx funkce k vytvoření seznamu v paměti, existujících hodnot a klíčů v registrech. Dále nezávisle na souborovém systému, rekurzivně prohledává složky od nevyšší až po nejnižší a vytvoří seznam v paměti. Použije k tomu FindFirstFile a FindNextFIle funkce. Pro každý soubor jsou zaznamenány údaje o velikosti v bajtech a souborové vlastnosti jako: hidden, systém archived a podobně. Pote je ještě zaznamenán čas posledního zápisu do souboru. Po zaznamenání druhé „snapshotu," Regshot porovná změny mezi jednotlivými „snapsohoty." Pokud dojde k vytvoření, upravení nebo smazání klíčů, hodnot nebo souborů registrů, dojde vytvoření poplachu a „changelogu."

Zpráva vytvořená Regshotem, se zakládá standartně ze čtyř částí:

1. Změny v registrech: Malware mění NoFolderOption nastavení v registrech, který zabraňuje uživatelům měnit, jak Windows Explorer zobrazuje soubory. Může také měnit DisalbeRegistryTools vlastnost nastavení, které zabraňuje uživatelům spouštět základní editor registrů (Regedit). Tím pádem uživatelé nemohou upravovat nebo odstranit záznamy v registrech provedené malwarem.
2. Přidané soubory: Malware vytvoří soubor csrssc.exe a uloží ho dočasných souborů uživatele. Tím vzniknou dva nové soubory v paměti. Ty to soubory jsou nepřímím výtvorem malwaru, protože tyto soubory vytvořil operační systém Windows nikoliv malware. Tento soubor se dá použít pro analýzu, jelikož jak můžeme vidět podle časové osy soubory 944983.exe a csrssc.exe běžely ve stejnou dobu, kdy běžel malware. Bez souboru 944983.exe můžeme pouze říct, že soubor csrssc.exe byl vytvořen, ale nemůžeme říct, zdali byl spuštěn.

3. Smazané soubory: Malware smaže soubor 944983.exe z počítače. Tento soubor je původním zdrojem viru. Tím pádem můžeme soudit, že malware po tom, co vykonal svoji činnost se sám smazal.
4. Upravené soubory nebo jejich vlastnoti: Malware nemění přímo žádné soubory. Všechny mění takzvaně nepřímo. Použije k tomu jinou API nebo program, třeba pokud by chtěl změnit historii tak to udělá skrze WinINet API, která automaticky změní index.dat soubor (Internet Explorer soubor historie).

4 Metodologie

V této kapitole bych pospal proces, který budu popisovat během praktické části. Budu se snažit postupovat popořadě, jak zde uvede postup. Budu je uvádět v jednotlivých podkapitolách.

4.1 Určení druhu škodlivé kódu

Pokud se chceme věnovat analýze škodlivého softwaru, tak se nevyhneme tomu, abychom se seznámili s tím, co to vlastně malware je, jaké může mít formy a jaké má kategorie. Předpokládám, že asi nikdo si nestáhne náhodně zavirovaný počítač a bude se v něm snažit hledat malware. Mnohem lepé se nám bude pracovat a učit se, pokud si určíme, s jakým typem malwaru se potkáme a zjistíme si jeho klíčové vlastnosti.

Pro tuto práci jsem si vybral 3 hlavní skupiny malwaru a to: Adware, Spyware, Trojské koně. Rád bych na tomto vzorku prezentoval jak změny provedené malwarem v daném sandboxu, tak i internetovou komunikaci mezi malwarem
a vzdáleným serverem.

4.2 Vytvoření analytického prostředí

Jednou z nejzákladnějších věcí, pokud se chceme věnovat jaké koly práci se škodlivým kódem. Připravím si více virtuálních prostředí [19], pro porovnání napáchaných škod malwarem, ale o tom později. Popíši, jak si připravit virtuální prostředí jak na platformě Windows 10, tak i na Fedoře, jakož to zástupce pro linux base operační systémy. Chtěl bych hlavně vytvořit řešení za pomoci skriptů, nechci se do hloubky věnovat prací s grafickým prostředím pro jednotlivé virtualizační nástroje, jelikož mi to přijde nezajímavé a nemyslím si, že by to obohatilo tuto práci. Pro operační systém Windows 10 jsem zvolil Hyper-V a pro Fedoru Virt a jeho přidružené nástroje, a to hlavně díky jejich podpoře automatizace a možnosti pracovat přes příkazový řádek. V bakalářské práci poté přidám automatickou instalaci vybraných analytických nástrojů. Pro umožnění internetové komunikace malwaru, budu muset navrhnou vlastní síť s infikovanými zařízeními a jedním analytickým. To bude sloužit jako „falešné “ internetové připojení, bude simulovat odpovědi vzdálených serverů a dotazy na

mě zaznamenávat do textového souboru. Pokud chceme používat nástroje zaměřené na porovnávání změn, tak je musíme spustit ještě před infikováním snadboxu, abychom neměly zkreslené výsledky.

4.3 Výběr analytických nástrojů

Důležitá je správná volba nástrojů a pomůcek. Musíme si také stanovit cíle analýzy a pozorování. Pokud si chceme udělat přehled o malwaru se kterým se potýkáme, je dobré ještě před samotnou analýzou nechat daný počítač prověřit nějaký antivirový program. Jelikož nám může poskytnou užitečná data, pokud se jedná o již známí malware. Dříve jsem mluvil o několika nástrojích, které se nám budou velice hodit.

Pro pozorování změn v registrech, použijeme program Reghot nebo Process monitor. Pro vytovření a směrování síťového provozu budu potřebovat použít dva nástroje Burp a INetSim. Dále také slouží k zaznamenávání internetovou komunikaci a její detaily. Pokud budu chtít monitorovat provoz na síti, nejpřístupnější program je Wireshark. Díky němu mohu mít přehled i o jednotlivých packetech a jejich obsahu, cílové adrese a protokolu, který používají. Program RegFsNotify nám dá přehled o změnách struktuře adresáře, umožní identifikovat změny provedené ve vlastnostech složek, jejich úpravu, mazání a vytváření. Pro analyzování knihoven použitých malwarem použiji Dependency Walker. Pokud budu chtít daný malware podrobit hlubší analýze za pomoci assembleru, tak první musím zjistit, jestli je daný soubor šifrován popřípadě komprimován. Na o využiji program RDG Packer Detector.

4.4 Infikování virtuálního počítače a provedení analýzy

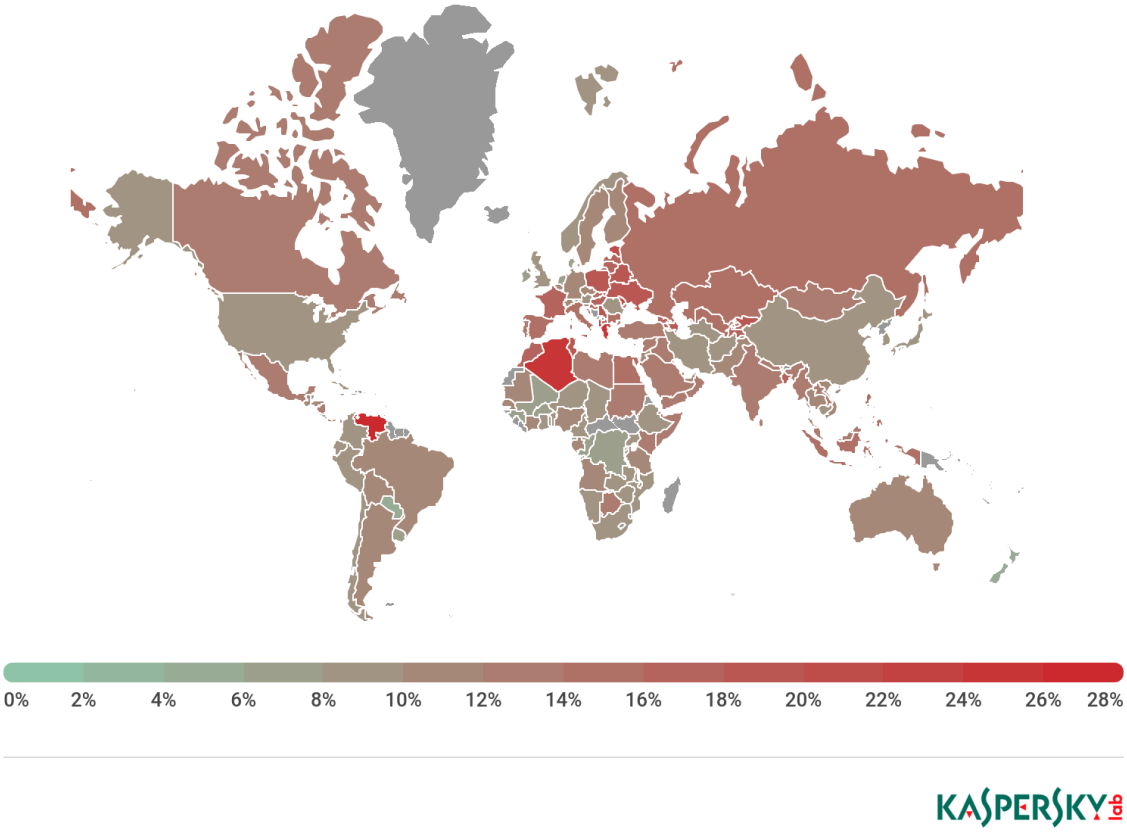
Po provedený všech příprav, nás teď čeká to nejzajímavější. Infikování sandboxového stroje, provedeme spuštěním buď infikovaných aplikací nebo přímo spuštěním malwaru. Teď nás čeká opětovné spuštění antivirových a dalších analytický programů. Pokud jsem, předtím provedli záznam původní stavu sandboxu, tak je nyní můžeme porovnat. Dále se můžeme podívat na výsledek antivirového programu. Pokud našel stejný nebo podobný malware, která již má v databázi, tak nám napíše jeho název a velikost jeho hrozby. Tomuto tématu se dále budu věnovat v bakalářské práci.

4.5 Vyhodnocení výsledů analýzy

Po shromáždění dostatku dat z naší analýzy a výsledků antivirových řešení, můžeme většinou jednoznačně určit, důsledky a škodu napáchanou malwarem. Z toho také můžeme vyvodit kroky co můžeme podniknout, abychom infikování zabránili. Může se stát, že si něčeho nevšimneme nebo že to vyhodnotíme špatně i to patří k této činnosti.

5 Bezpečností rizika způsobená škodlivým kódem

Již dříve jsem již zmínil o globálním dopadu malwaru a nákladech spojených s jeho bojem. Nyní se budu věnovat jeho druhům, o kterých se zmiňuji ve své práci. Týká se to konktrétně Trojských koňů, spywaru a adwaru. Jelikož jejich hrozby jsou stále aktuální a ovlivňují celou moderní společnost. Největší hrozba u těchto skupin, představuje jejich možnost nepozorovaného účinkování na napadeném počítači. Mapa níže, ukazuje zemně od největšího risku napadení počítače skrze nakažení přes síť. Od červené, kde je pravděpodobnost největší (Venezuela 29,76 %) až po barvu zelenou, které představuje relativně bezpečné zemně. Tyto údaje byly nashromážděny institucí Kaspersky Lab [20].



Obr. 5.1: Mapa zobrazující nebezpečí infekce přes síť v jednotlivých zemích [20]

Oběti útoku malwarem můžeme rozdělit do dvou největších skupin, útoky na firmy a útoky na soukromé osoby. Toto bych tu chtěl ukázat porovnáním dvou tabulek, a to počtu detekovaných napadení mezi roky 2017/2018 [21], seřazeno od největší hrozby, co se do počtu útoků týče. První tabulka se věnuje útokům na firmy, zde můžeme vidět ohromný nárůst útoků Trojským koněm, Backood, Spyware a Riskware Tool. To je i důvod proč jsem si právě vybral tyto skupiny na analýzu. Jiné typy útoků jsou zase na ústupu, což je normální cyklus, jelikož se útočníci vždy soustředí na útoky, které jim přinesou největší prospěch. U soukromých osob, můžeme pozorovat, že jako nejrozšířenější útok je považován za adware. Pokud porovnáme počet napadení mezi jednotlivými tabulkami, tak útoků na soukromé osoby je přibližně 10 krát více, dále si můžeme povšimnou, že počet útoků na firmy stoupl mezi lety 2017/2018 téměř o 80%.

Detekované útoky na firmy		
Pozice	Typ útoku	R/R% změna
1	Trojský kůň	132
2	Hijacker	43
3	Riskware tool	126
4	Backdoor	173
5	Adware	1
6	Spyware	142
7	Ransomware	9
8	Červ	-9
9	Rogue	-52
10	Hack tool	-45
Celkový počet detekovaných útoků		
2017	39970812	
2018	71823114	79%

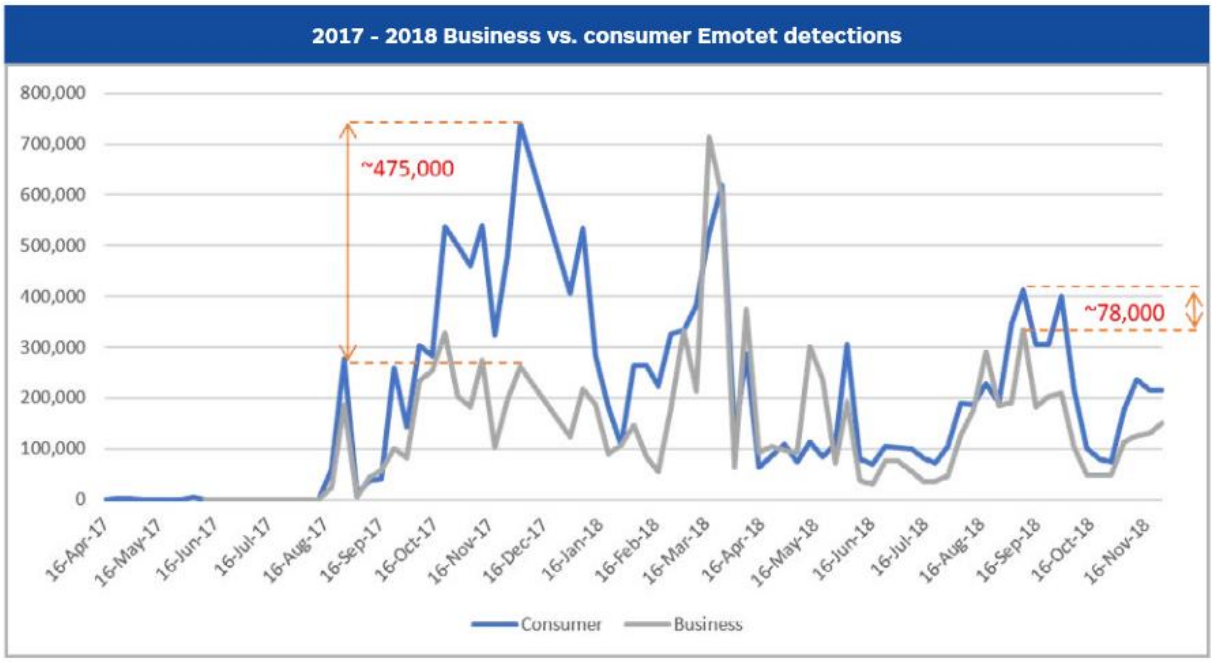
Tab. 5.1:Tabulka obsahující typy útoků na firmy podle jejich závažnosti

Detekované útoky na soukromé osoby		
Pozice	Typ útoku	R/R% změna
1	Adware	-39
2	Trojský kůň	19
3	Riskware tool	7
4	Backdoor	34
5	Adware	-36
6	Hijacker	-84
7	Červ	-28
8	Spyware	27
9	Ransomware	-29
10	Rogue	-39
Celkový počet detekovaných útoků		
2017	775327346	
2018	750296307	-3%

Tab. 5.2: Tabulka obsahující typy útoků na soukromé osoby podle jejich závažnosti

V posledních letech byly zaznamenány dva hlavní trojské koně, které představují veliké riziko hlavně pro firmy, ale také pro soukromé osoby. Jsou hlavně zaměřené na získávání citlivých informací a mohou se dále rozšiřovat. Emotet se postupně vyvíjel, z roky jeho vývoje se také rozrůstaly jeho účely a cíle. Hlavním účelem je odcizení dat z počítače oběti, dokáže také odposlouchávat síťovou komunikaci a nově spolu šíří také trojské koně zaměřené na bankovní sektor. K šíření používá mimo jiné i slabinu EternalBlue. Druhým

malware se jmenuje TrickBot. Je to takzvaný „bankovní trojský kůň," dále také krade uživatelská data a kradení kryptoměn z elektronických peněženek. Ve svém cíli je jinak hodně podobný právně s výše zmíněným Emotetem. Graf níže nám ukazuje, jak za poledni dva roky došlo k rozšíření a napadení ohromného množství uživatelů za velice krátkou dobu. Tmavě modrá čára představuje křivku počtu napadení zařízeních na zařízeních vlastnících soukromé osoby, šedá křivka zase představuje útoky na firmy. Jak můžeme vidět, od srpna do konce roku se prakticky z nulového počtu napadených zařízení vyšplhalo až přes závratných 700 000 napadení. Samozřejmě tu mluvíme o napadení chráněného počítače, jelikož došlo k detekci útoku antivirovým programem. Kolik strojů podlehlo, co nebyly řádně zabezpečené můžeme jenom odhadovat. Dále je také zajímavé, že počet napadených firem je v některých obdobích výrazně menší než u soukromých osob. Z toho bychom mohli usoudit, že soukromé osoby nedbají tolik na zabezpečení svého zařízení, nebo jenom kvůli mnohem většímu počtu strojů obecně.



Obr. 5.2: Graf porovnávající počet napadených soukromých a firemních strojů útokem Emotet [21]

6 Jak posupovat při vytváření sandboxu

Všeobecně platí, že mám dvě možnosti, jak postupovat při vytváření virtuálních strojů. Zaprvé mohu použít nástroj s GUI (graphical user interface) – grafické uživatelské prostředí. Kdy mohu přehledně a jednoduše vidět všechno podstatné, a někdy mi pomůže nápověda. Druhá možnost je trochu těžší pro lidi, co nemají zkušenosti s konzolí. Jelikož se prování právě přes příkazový řádek, jak ve Windows (Powershell), tak pro Fedoru (Shell). Používání konzole, mi pomůže při možné automatizaci či vytváření různých scriptů. Některé příkazy tu v dále budu ukazovat a popisovat, jak fungují.

6.1 Volba operačního systému

Cílem této kapitoly, je vybrání a odůvodnění operačního systému, na kterém budu poté připravovat virtuální stroj. Zmíním tu virtualizační nástroje, které na jednotlivých platformách použiji a důvod proč jsem si je vybral.

6.1.1 Windows 10

Jako první se budu věnovat systému Windows 10, jelikož je to operační systém, na kterém trávím nejvíce času a každý den ho aktivně používám. Dále si myslím, že větší skupina uživatelů, laiků bude používat právě ten tento systém, díky jeho rozšířenosti a přehlednému a jednoduchému ovládání.

Jak jsem psal víše, prvně popíšu způsob, jak vytvořit a použít virtuální stroj na Windows 10 s Hyper–v. Jelikož modul Hyper-v je možný aktivovat až s edicí Windows 10 Pro, Enterprice nebo Education a jelikož mám na počítači jenom Windows 10 Home Edition, tak jsem se rozhodl k takové kličce. A to že si budu virtualizovat Windows 10 Pro ve Virtual Boxu a poté v něm budu dále pracovat. Toto řešení není úplně optimální, jelikož virtualizací virtualizovaného stroje ztrácím podporu pro hardwarovou akceleraci. K tomuto rozhodnutí mě dovedlo hlavně uvážení, že bych chtěl ukázat práci s powershellem, a tím pádem je pro mě nejlepší volbou Hyper-v.

6.1.2 Fedora

Tímto operačním systémem by chtěl reprezentovat skupinu lidí, co analyzování škodlivého softwaru dělají více do hloubky, nebo prostě preferují operační systémy ze skupiny linux. Mám ho nainstalovaný na notebooku Lenovo. Pro tuto platformu jsem zvolil možná trošku netradiční, ale pro mě známý virtualizační nástroj Virt [22]. Virt patří do skupiny nástrojů libvirt. Vytváří KVM nebo Linux kontejnery. Pro distribuce linuxů se dají stáhnou samozřejmě i známější virtualizační nástroje jako: VirtualBox, VM Ware. Jedná se o nástroj určený pro enterprice prostředí, takže občas působí neohrabaně a nejednoznačně. Díky funkci virt-custom nebo virt-manage můžeme modifikovat daný image, ještě před tím, než ho vůbec spustím. Tím si mohu automaticky připravit prostředí a nainstalovat si programy, které budu dále používat jako třeba: Wireshark, Process monitor a Regshot.

7 Příprava sandboxu

V této kapitole popíšu, jak si připravit virtuální prostředí na platformách Windows 10 a Fedora. Hlavně bych se zaměřil na předvedení příkazů v prostředí powershell a shell. Provádění instalace přes grafické prostředí zde zmíním jenom okrajově, jelikož je to postup, který je již dobře zdokumentován. Poté popíšu, jaké nástroje si připravím, kde je získám a jak je naistaluji. Na závěr představím a popíšu, jak vytvořit a připravit vlastní síť se simulovaným internetovým připojením. Budu k tomu potřebovat ještě jeden virtuální počítač, Ubuntu. Předvedu, jak nastavit statické IP adresy a přesměrovat komunikaci na jedno zařízení. Vlastní síť budu potřebovat, abych mohl bezpečně zkoumat komunikace malware se vzdálenými servery.

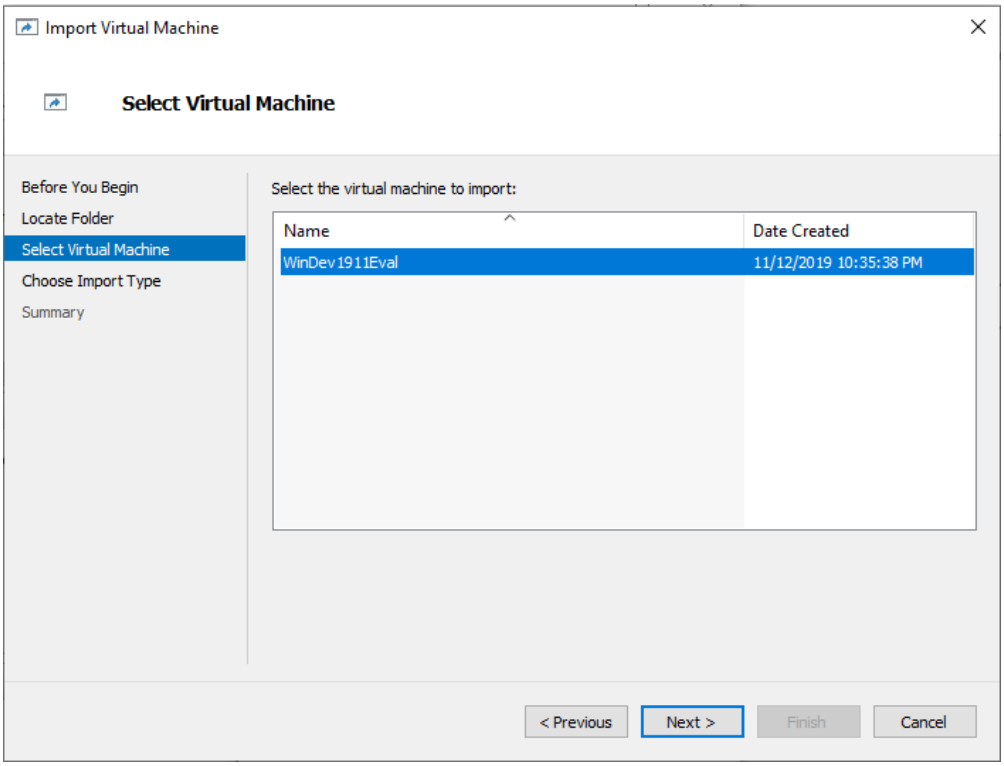
7.1 Vytváření virtuálního stroje

Budu se věnovat postupu při vytváření virtuálního stroje na daných systémech za pomoci grafické rozhraní. Následně předvedu příkazy pro powershell a bash, které poslouží později k plné automatizaci.

7.1.1 Windows 10

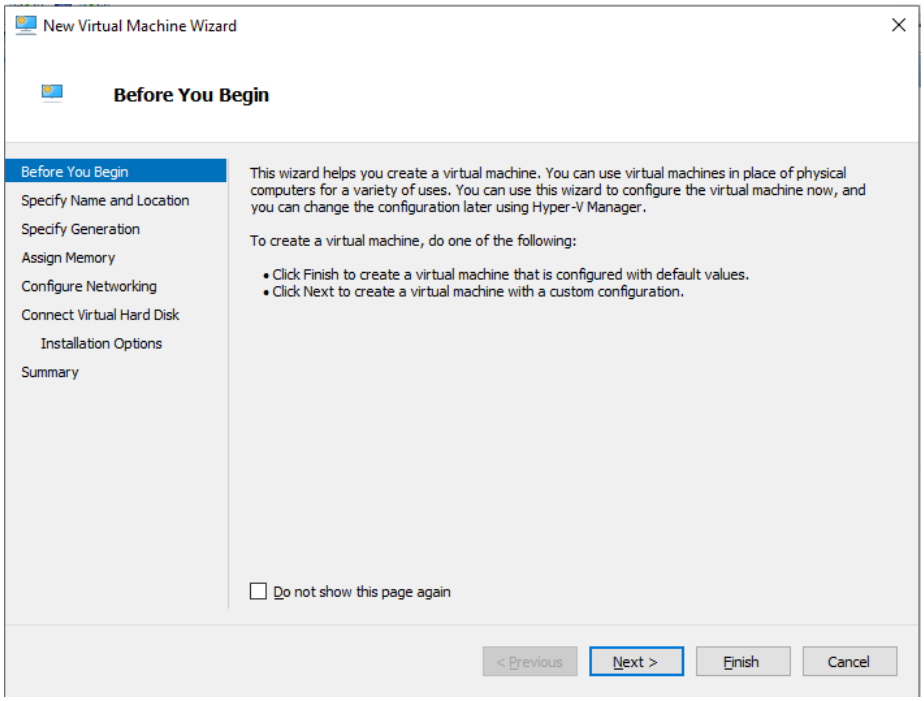
Vytváření virtuálního stroje za pomocí grafického rozhraní je poměrně jednoduché, proto mu nebudu tady věnovat tolik času, více se poté budu věnovat práci s konzolí. Chtěl bych zde ukázat pár příkazů a poté v bakalářské práci to rozšířit na script, který mi připraví virtuální stroj.

Pokud se mi nechce virtuální stroj nějak zásadně upravovat, mohu si stáhnou již připravený obraz pro jednotlivé virtuální nástroje, jsou to soubory s příponou .VHDX. Poté jednoduše spustím Hyper-v managera. Zapne se mi vybraný virtualizační program. Zobrazí se mi okno, viz **Chyba! Nenalezen zdroj odkazů.**. 8.1. Vidím zde jednotlivé virtuální počítače vhodné pro import. Dále si zde vyberu, jestli chci naimportovat do stejné složky nebo celý virtuální stroj nainstalovat jinam. Poté stačí zmáčknout už jen tlačítko Finish. Tím dokončím proces importování a po dokončení procesu si už mohu daný virtuální stroj zapnout skrze správce.



Obr. 7.1: Import virtuálního stroje ze souboru .vhd

Za druhé, mohu použít instalační disk nebo médium, soubor .iso. Tento obraz si také mohu stáhnout z oficiálních webových stránek nebo pokud jde o platformu Windows, tak si mohu vytvořit díky nástroji Media Creation. Otevřu si Hyper-V manager, vyberu položku New -> Virtual Machine… Následně se mi otevře instalační okno, viz. (Obr. 7.1). Na první stránce kliknu na tlačítko Next, to mě posune na část, kde si zvolím název a možné umístění virtuálního stroje. Tlačítkem Next se přesunu na stránku, kde si můžu zvolit jakou generaci virtuálního stroje chci. Jde zde o volbu mezi starším, který podporuje 32-bitové operační systémy nebo modernější s UEFI firmwarem a 64-bitovou verzí. Ve čtvrté kartě si zvolím množství alokované paměti pro virtuální stroj. Bylo by dobré alokovat minimálně 2048 kB paměti pro každý virtuální stroj. Další záložka obsahuje nastavení síťového připojení. Mohu zde vybrat nepřipojeno nebo default switch. Po nastavené sítě mě čeká vložení virtuálního disku nebo připojení instalačního media. Pokud nemám virtuální disk, tak ho zde vytvořím. Dále je zde možné zvolit jeho umístění a velikost. V podzáložce si vyberu, z jakého instalačního média chci instalovat a zadám jeho cestu. Poté se mi už zobrazí jenom přehled nastavení a tlačítkem Finish dokončím instalaci.



Obr. 7.2: Instalační průvodce Hyper-V

Vytvoření powershellového skriptu ukážu v bakalářské práci, teď se budu věnovat pouze předvedení příkazů, které použiji a jejich popisu. Proč jsem je zvolil a jaký mají pro nás význam. Samotný skript se bude skládat z více částí. Za prvé stažení daného Hyper-V obrazu. Za druhé část, kde se nám virtuální stroj nainstaluje. Jako poslední bych se chtěl pokusit o automatické nainstalování potřebných programů například: Wireshark. Teď už k samotnému stahování.

Nejjednodušší příkaz, stejný jako pro Linux je wget. První argument zde je URL adresa stahovaného souboru, tedy odkaz na Windows 10 Hyper-V obraz. Outfile argument určuje jméno a cestu k souboru, kam se daný obraz stáhne.

Výpis kódu 7.1: Stažení obrazu Windows 10 pomocí příkazu wget

```
wget "https://aka.ms/windev_VM_hyperv" -outfile "C:\Work\Virtual_machine"
```

Pokud chci využít „PowerShell 3“ tak mohu pro stahování použít příkaz Invoke-WebRequest. Tento proces má více použití než wget, protože umožňuje nejen stahovat, ale i analyzovat soubory. V ukázce příkazu níže mohu vidět již parametrizovaný příkaz, tedy URL adresu a cílový soubor nezadáváme přímo do příkazu, ale využívá k tomu proměnné $url a $file_output. Parametr -Uri určuje danou URL adresu a -OutFile zase jméno a cestu k souboru [23]. Tuto formu příkazu použiji poté ve vytvářecím scriptu.

Výpis kódu 7.2: Stažení obrazu Windows 10 pomocí příkazu Inwoke-WebRequest

```
$url = "https://aka.ms/windev_VM_hyperv"
$file_output = "C:\Work\Virtual_machine"

Invoke-WebRequest -Uri "$url" -OutFile "$file_output"
```

Soubor se mi stáhne ve formátu .zip, takže jednoduchým příkazem Expand-Archive extrahuji daný zip. Uvedu cestu k .zip souboru a cílovou složku, pro mě C:/Work.

Výpis kódu 7.3: Rozbalení archivovaného obrazu Windows 10 pomocí příkazu Expand-Archive

```
Expand-Archive  C:\Work\WinDev1911Eval.HyperVGen1.zip  -DestinationPath
C:\Work
```

Teď mě čeká samotné vytvoření virtuálního stroje. Pro získání podrobných informací, týkající se příkazů nebo příklady mohu využít buď oficiální dokumentace, nebo nápovědy v samotné konzoli powershellu, kterou spustíme příkazem.

Výpis kódu 7.4: Zobrazení nápovědy za pomoci příkazu Get-Help

```
Get-Help Get-VM
```

Ještě před samotným vytvořením, musím udělat virtuální přepínač. Přepínač bude v interní síti, tudíž se bude nacházet na rozmezí 192.168.x.x nebo 172.16.x.x až 172.31.x.x nebo 10.x.x.x.. Tyto rozsahy sítí se nepřesměrují do veřejného internetu. Z kódu níže je asi nejzajímavější příkaz „$net.Name“, který zobrazí jméno aktuálně používaného internetového adaptéru. Jinak se zase snažím parametrizovat všechno podstatné. Je možné, že některé parametry se budou opakovat, což je problém jen teď, jak ukazuji jednotlivé kousky. Celistvý script by samozřejmě obsahoval každý parametr jednou.

Výpis kódu 7.5: Vytvoření virtuálního přepínače pomocí příkazu New-VMSwitch

```
$SwitchName = "Virt-Switch"
$SwitchType = "Internal"
New-VMSwitch   -SwitchName   "SwitchName"   -SwitchType   Internal   -
NetAdapterName $net.Name
```

Pro samotné vytvoření použiji příkaz New-VM. Vložím zde již hotový parametrizovaný příkaz. Umožňuje nastavovat 5 základních proměnných:

1. Jméno virtuálního stroje
2. Množství alokované RAM paměti
3. Vesta k VHD souboru
4. Cesta kam se virtuální stroj nainstaluje
5. Jméno virtuálního směrovače, na který se připojí

Výpis kódu 7.6: Parametrizovaný příkaz pro vytvoření virtuálního stroje Windows 10

```
$VMName = "Windows-10-Pro"
$MemoryBytes = "4GB"
$VHDPath = "C:\Work\Virtual-machine\WinDev1911Eval.HyperVGen1\Virtual-
Hard-Disks\WinDev1911Eval.vhd"
$PathToVm = "C:\Virtual Machines\$VMName"
$SwitchName = "Virt-Switch"

 $VM = @{
     Name = $VMName
     MemoryStartupBytes = $MemoryBytes
     Generation = 1
     VHDPath = $VHDPath
     Path = $PathToVm
     SwitchName = $SwitchName
 }

 New-VM @VM
```

Tak a teď mám vytvořený virtuální počítač s operačním systémem Windows 10 a připojený k internímu přepínači. Dále stačí jenom virtuální stroj zapnout příkazem.

Výpis kódu 7.7: Spuštění virtuálního stroje Windows 10

```
$VMName = "Windows-10-Pro"
 Start-VM -Name "$VMName"
```

7.1.2 Fedora

Na platformách linux s grafickým prostředím, probíhá instalace hodně podobně, pokud si zvolím virtualizačního managera s grafickým prostředím například: VirtualBox. Já by tu rád ukázal způsob trošku komplikovanější za použití příkazového řádku a programovacího jazyka bash.

Rád bych zde ukázal jednotlivé příkazy, následně bych je implementoval do jednotlivých funkci a poté bych je zde popsal a jejich funkci. Prvně si musím stáhnout obraz virtuálního systému ve formátu .ova. Pokud by se jednalo o linux distribuci, mohu použít i „raw“ formáty například: qcow2 a raw.. Zde pro stažení použiji příkaz wget. Jeho podoba je velice podobná jako pro powershell. Použiji přepínač „--progress=dot:giga“ díky němu mi do konzole budu přicházet výpis po každém „Gb“ dat nikoliv po každém „Kb.“ Tím zredukuji přehlcení okna konzole a zároveň zachovám možnost sledovat postup stahování.

Výpis kódu 7.8: Stažení obrazu Windows 10 pomocí příkazu wget

```
$url = "https://aka.ms/windev_VM_hyperv"
$file_output = "/home/Fedora/Work/"

wget -c ${url} -O ${file_output} --progress=dot:giga
```

Pro extrahování souboru, použiji program unzip. Ten si předtím nainstaluji z RPM. Používám příkaz „sudo“ protože z bezpečnostního hlediska nepoužívám uživatele Root. Pokud do příkazu nedám cestu, kam má provést extrahování, tak se provede do aktuální složky.

Výpis kódu 7.9: Rozbalení archivovaného obrazu Windows 10 pomocí příkazu unzip

```
ZipPath = /home/Fedora/Work/WinDev1911Eval.HyperVGen1.zip
sudo yum install -y unzip
unzip ${ZipPath}
```

Vytvářecí příkaz pro tento příklad je trošku komplikovanější, chci provést podrobnější nastavení. Jedna si zde určuji počet jader virtuálního počítače, dále přidělím velikost ram paměti. Přepínačem „--os-type“ určujeme o jaký druh operačního systému se jedná a „--os-variant“ specifikuje, že jde o Windows 10. Pro připojení.ova image je zde „--disk“. Sada přepínaču „--nographics --noreboot –noautoconsole “ tu zastává roly pro moje budoucí upravování obrazu. Vysvětlím, co dělají od prvního do posledního: „--nographics“ nedovolí

grafické prostředí při instalaci a nepřeruší tím moji běžící sezení v konzoli „—noreboot“ jak už asi název napovídá, nedovolí po provedení instalace automatické spuštění virtuálního stroje. Jako poslední přepínač „--noautoconsole“ zamezí ve snaze se automaticky připojit k nově vytvořenému virtuálnímu stroji. Tato sado přepínačů by neměla vypisovat žádné chybové hlášky, ty mohou nastat, pokud vynechám poslední z nich. Poté tu dvě pomocné přepínače „--debug --serial pty“ ty teoreticky ve finálním scriptu nepotřebujeme, ale pro ladění a řešení možných chyb se hodí [22].

Výpis kódu 7.10: Vytvoření virtuálního stroje Windows 10 v jazyce bash

```
VMName = "Windows-10-Pro"
GUEST_CPUS = "2"
GUEST_RAM = "4096"
OvaPath  =  "C:\Work\Virtual-machine\WinDev1911Eval.HyperVGen1\Virtual-
Hard-Disks\WinDev1911Eval.ova"


function create_vm() {
    virt-install --name ${VMName} \
        --cpu host --vcpus ${GUEST_CPUS} --ram ${GUEST_RAM} \
        --os-type=windows --os-variant=windows10 } \
        --disk ${OvaPath} --import \
        --nographics \
        --noreboot --noautoconsole \
        --debug --serial pty
}
```

Pro nastartování a připojení se do daného virtuální prostředí použiji dva příkazy. Musím prvně nastartovat virtuální stroj, jelikož jsem mu to zakázal při jeho vytváření, poté se jenom připojím do konzole. Ta se otevře v naší, tudíž dojde k přerušení všech příkazů nebo scriptů.

Výpis kódu 7.11: Spuštění virtuálního stroje Windows 10 s virsh managerem

```
VMName = "Windows-10-Pro"

virsh start domain ${VMName}
virsh console domain ${VMName}
```

Pro teď toto stačí. Jelikož mám připravený virtuální stroj, v bakalářské práci tu ještě popíšu způsob, jakým naistalovat programy na obraz stroje bez jeho spuštění, jeho zachycení a nahrání.

7.2 Instalace nástrojů pro analýzu škodlivého kódu

Je zde mnoho různých nástrojů, od monitorovacích nástrojů systému Windows, až po specializované aplikace pro analýzu spustitelných souborů exe. Budu tu hlavně ukazovat nástroje o kterých jsem dříve mluvil.

Všechny nástroje budu stahovat z jejich oficiálního zdroje, což po většinou bude jejich webová stránka, některé by se možná daly najít i na „Windows store“ nebo v RPM na Fedoře. Určitě bych zde chtěl mít alespoň 1 antivirový program. Což bude určitě Windows defender, jelikož je součástí každého nového operačního systému od Microsoftu, mohlo by být zajímavé jeho srovnání s nějakým antivirovým programem z konkurenční firmy jako například AVAST. Ten stáhnu z jejich webu, pravděpodobně ve formátu spustitelného souboru exe. Pro monitorování síťového provozu budu používat nástroj Wireshark, pro instalaci na platformě Windows musíme instalovat skrze oficiální instalační soubor stažitelný z jejich internetové stránky, pro platformu linux je program dostupný skrze instalaci přes RPM. Program INetSim, který budu používat na simulování internetových odpovědí nainstaluji na linux přes RPM balíčky.

Některé nástroje nepotřebují instalaci, jako třeba Regshot, RDG Packet Detector a Process Monitor. Tyto nástroje stáhnu a pravděpodobně budou v komprimovaném formátu zip, rar, pro linux gz, tar. Po jejich stažení je jenom rozbalím do požadované složky a poté je mohu rovnou zpustit.

Nástroje, které mohu použít pro platformu Windows nemusí a pravděpodobně nebudou fungovat pro platformu linux. Tudíž musíme zvolil, jak správné nástroje pro jednotlivé platformy. Jelikož tu zatím řeším hlavně Windows 10, tak i nástroje které jsem zde uváděl jsou určeny pro tuto platformu.

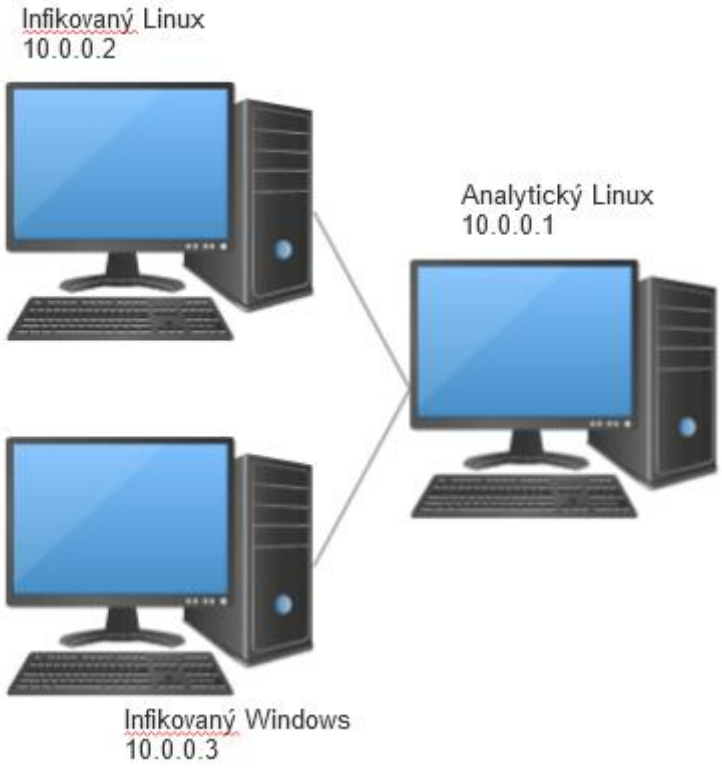
7.3 Připravení simulovaného internetového připojení

Abych mohl analyzovat škodlivý software, který využívá přístupu k internetu, musím si vytvořit vlastní internet. Jeho nastavení je komplikovanější a využívá hned několik virtuálních strojů. K tomu využiji nástroj zvaný INetSim, který mi umožní simulovat odpovědi z Internetových služeb. Využívá internetové protokoly HTTP, DNS, SMTP a další. Abych mohl simulovat protokol SSL budu využívat program Burp. To mi umožní za běhu generovat SSL certifikáty pro můj

infikovaný virtuální počítač. Burp také využiji k vytvoření transparentního SSL proxy serveru, který se bude nacházet mezi mojí infikovanou virtuální jednotkou a analytickým INetSim virtuálním počítačem. Pokud nepotřebuji sledovat SSL provoz tak program Burp vůbec nepotřebuji. Pokud chci sledovat šifrovanou komunikaci, kterou škodlivý kód může komunikovat, tak můžu pro můj infikovaný počítač vytvořit a nahrát single root CA certifikát.

7.3.1 Nastavení analytické sítě

Zde využiji dříve připravené stroje, jen je uspořádám v rámci separátní sítě viz diagram (Obr. 7.3).



Obr. 7.3: Diagram virtuální sítě

Jelikož budu používat nástroj INetSim, tak jako analytický počítač musím použít Linux distribuce Ubuntu, jelikož program je napsán pro tuto verzi. Dále by měl fungovat i na všech derivacích Ubuntu jako je Debian a další. Ubuntu si vytvořím stejně jako Fedoru akorát použiji jiný iso popřípadě ova zdrojový soubor. Jako první si nastavím „Analytický Linux.“ Kromě nainstalování běžných věcí, potřebuji nastavit statickou IP, to můžeme u distribuce Ubuntu udělat přes upravení souboru /etc/network/interfaces, kam na konec souboru přidám.

Výpis kódu 7.12: Nastavení statické IP v souboru interfaces

```
auto enp0s3
iface enp0s3 inet static
 address 10.0.0.1
 netmask 255.255.255.0
```

Toto nastaví statickou IP adresu 10.0.0.1 a příslušnou masku k síťovému adaptéru enp0s3. Inet static zajistí, aby IP adrese byla staticky přidělená, doté doby, dokud buď neodstraním nebo nezměním tento konfigurační soubor. Dále už jenom zapnu uvedené síťové rozhraní enp0s3 příkazem:

Výpis kódu 7.13: Zapnutí síťového rozhraní

```
$ sudo ifup enp0s3
```

Nastavení sítě mám tímto hotové. Teď nainstaluji programy Burp a INetSim. INetSim budu instalovat přes konzoli z repositáře za pomoci příkazu apt install. Je možné že nejprve budu muset přidat odkaz na daný binární soubor do source.list.d souboru. Celá instalace bude tedy probíhat asi těmito kroky.

Výpis kódu 7.14: Instalace programu INetSim

```
$ echo "deb http://www.inetsim.org/debian/ binary/" >
/etc/apt/sources.list.d/inetsim.list
$ sudo wget -O - http://www.inetsim.org/inetsim-archive-signing-key.asc
| apt-key add -
$ sudo apt update
$ sudo apt install inetsim
```

Nástroj Burp se bohužel nedá naistalovat přes balíčky z depozitáře, takže si ho musíme stáhnou z webové stránky „https://portswigger.net.“ Poté už jenom spustíme instalační script příkazem.

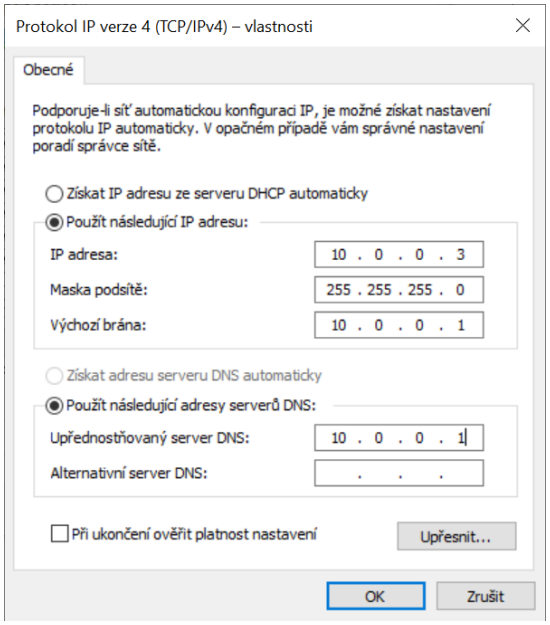
Výpis kódu 7.15: Instalace programu BurpSuite

```
$ sudo sh ~/Downloads/burpsuite_community_linux_v2020_2_1.sh
```

Burp se instaluje do adresáře „usr*/local/BurpSuiteCommunity.“* Tímto mi příprava toho stroje skončila.

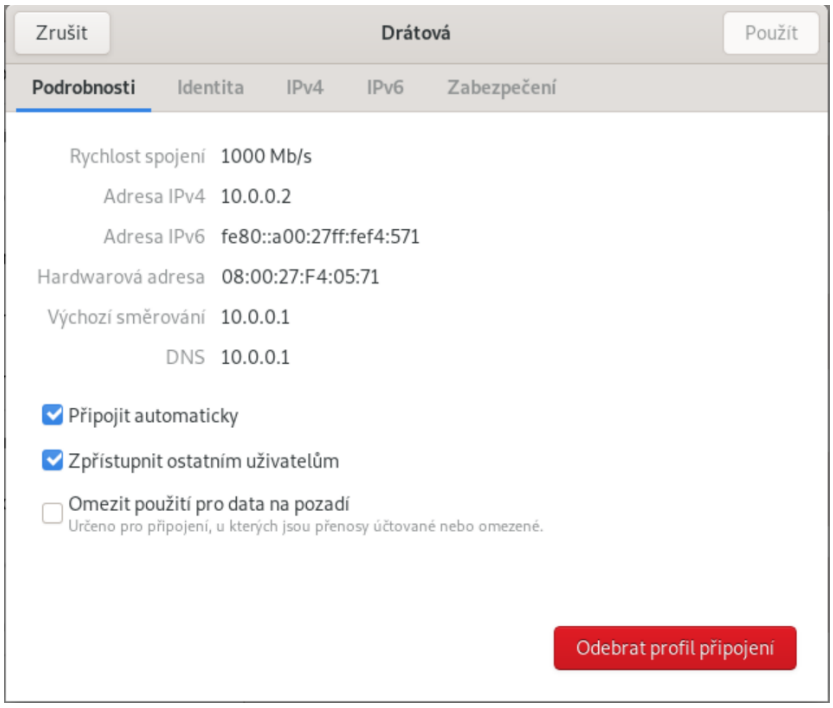
Nastavení sítě u Windows 10 je jednoduché. Otevřu si síťová připojení v „Ovládací panely\Síť a internet\Síťová připojení“ vyberu příslušný adaptér kliknu na vlastnosti, Protokol IP verze 4. Přidělím stanovenou IP adresu, masku, výchozí bránu a DNS server. Tím zajistím, že veškeré snahy komunikovat z toho

to počítače budou přesměrovány na můj Analytický počítač. Nastavení adaptéru je na obrázku níže (Obr. 7.4).



Obr. 7.4: Nastavení statické IP na systému Windows 10

U virtuálního počítače Fedora můžeme provést nastaveí jak přes grafické prostředí, tak i přes konzoli. Nastavím IP adresu na 10.0.0.2, výchozí bránu na 10.0.0.1 a DNS 10.0.0.1. Tím zase zajistím, aby síťová komunikace probíhala s mým analytickým počítačem.



Obr. 7.5: Nastavení statické IP na systému Fedora

Pokud bych potřeboval použít konzoli, tak použiji nástroj nmcli. Nejprve si musím zjistit UUID mého síťového adaptéru, to udělám příkazem:

Výpis kódu 7.16: Příkaz pro zobrazení informací o síťových adaptérech

```
$ nmcli connection show
```

```
[testsubject@localhost network-scripts]$ nmcli connection show
NAME                UUID                                  TYPE      DEVICE
Drátové připojení 2 d2ba026b-28b6-3ae1-b312-49988eeb2ed5  ethernet  enp0s8
System enp0s3       3c36b8c2-334b-57c7-91b6-4401f3489c69  ethernet  enp0s3
virbr0              0e63f1f5-4529-4450-8815-9617bbf9406b  bridge    virbr0
Drátové připojení 1 0dd2e375-a901-3e7f-baa6-23234999d4db  ethernet  --
```

Obr. 7.6: Zobrazení informací o síťových adaptérech

Poté pomocí příkazu nastavím IP adresu, síťovou masku a výchozí bránu:

Výpis kódu 7.17: Nastavení síťového adaptéru enp0s3

```
$ sudo nmcli connection modify „UUID“ ipv4.address „10.0.0.2/24 10.0.0.1“
```

Obr. 7.7: Ukázka příkazu pro zadání IP adresy síťovému adaptéru

Dále podobným příkazem nastavím DNS, přepíši „address“ na DNS a IP adresu na „10.0.0.1“.

Výpis kódu 7.18: Nastavení DNS pro síťový adaptér enp0s3

```
$ sudo nmcli connection modify „UUID“ ipv4.DNS 10.0.0.1
```

Po nastavení všech potřebných parametrů, restartuji síťový adaptér příkazem:

Výpis kódu 7.19: Restartování síťového adaptéru enp0s3

```
$ sudo nmcli connection reload „UUID“
```

Tím mám nastavenou síť mezi virtuálními počítači. Pro ověření funkčnosti postačí obyčejný příkaz ping, který mi ukáže, zda je možná komunikace mezi stroji. Pokud by příkaz z jakéhokoliv důvodu nefungoval, stačí provést nastavení znovu popřípadě, restartovat síťové adaptéry, aby jejich nastavení bylo aktuální.

7.3.2 Nastavení INetSim

INetSim umožňuje konfiguraci pomocí konfiguračního souboru, ten se nachází v této adresářové struktuře: „**/etc/inetsim/inetsim.conf.**“ Více se o konfiguraci píše v dokumentaci. Ukážu zde všechny potřebné změny, které mi umožní analyzovat případnou snahu o komunikaci škodlivého kódu.

Jelikož budu chtít měnit nastavení před novou analýzou, tak je nejjednodušší si program INetSim nakopírovat na místo kde, ho budu mí po ruce a kam se mi budou ukládat výsledky jednotlivých analýz.

Jako první si vytvořím adresářovou strukturu. Složku analysis a podsložku test-analysis, ve které budu mít konfigurační soubory. Dále překopíruji již zmíněný konfigurační soubor, a také sloužku data, která obsahuje vlastní program INetSim. Té poté upravím oprávnění 777.

Výpis kódu 7.20: Připravení programu INetSim před použitím

```
$ mkdir -p analysis/test-analysis
$ cp /etc/inetsim/inetsim.conf analysis/test-analysis
$ sudo cp -r /var/lib/inetsim analysis/test-analysis/data
$ sudo chmod -R 777 data
$ cd analysis/test-analysis
```

Jako další krok potřebuji nastavit konfigurační soubor programu INetSim, kde upravím následující hodnoty:

Výpis kódu 7.21: Nastavení konfiguračního souboru INetSim

```
#service_bind_address    10.0.0.1
Nahradím za
service_bind_address     0.0.0.0
```

Abych zabránil konfliktu DNS serverů, jelikož Ubuntu obsahuje jeden již ze základu, musím vypnout službu systemd-resolved.

Výpis kódu 7.22: Vypnutí základní služby DNS na Fedoře

```
$ sudo systemctl disable systemd-resolved.service
$ sudo service systemd-resolved stop
```

Výpis kódu 7.23: Nastavení přesměrování HTTPS v konfiguračním souboru INetSim

```
#dns_default_ip    10.0.0.1
#Stačí smazat #, tím uvedu nastavení v platnost
dns_default_ip    10.0.0.1

#https_bind_port 443
#Změním port protokolu HTTPS ze základného 443 na 8443
https_bind_port 8443
```

Nyní můžu zapnout program příkazem.

Výpis kódu 7.24: Zapnutí programu INetSim

```
$ sudo inetsim --data-dir=data --config=inetsim.conf
```

7.3.3 Nastavení Burp pro analýzu SSL

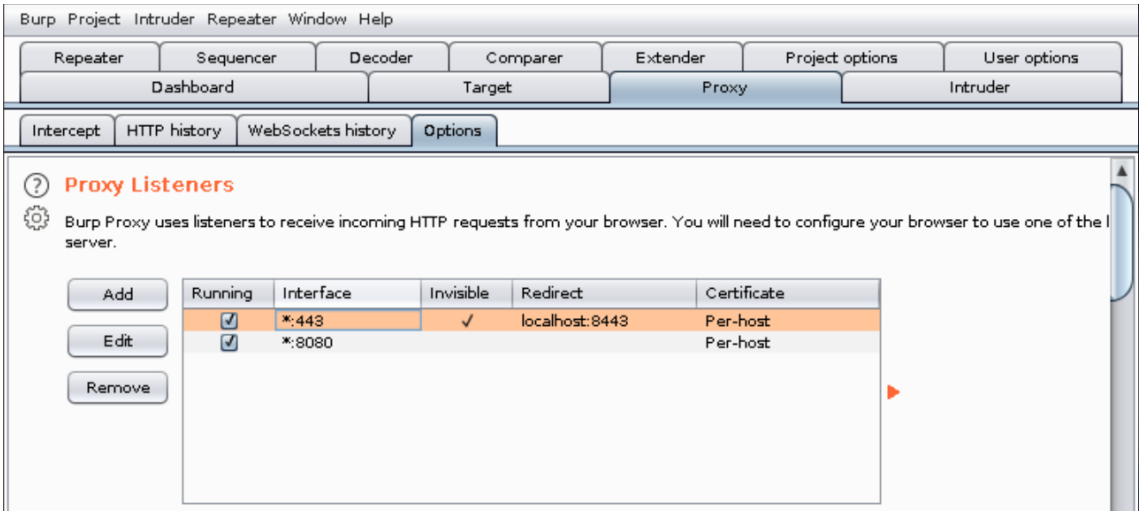
Burp spustím jako „super uživatel“ příkazem:

Výpis kódu 7.25: Spuštění programu BurpSuite

```
$ sudo /usr/local/BurpSuiteCommunity/BurpSuiteCommunity
```

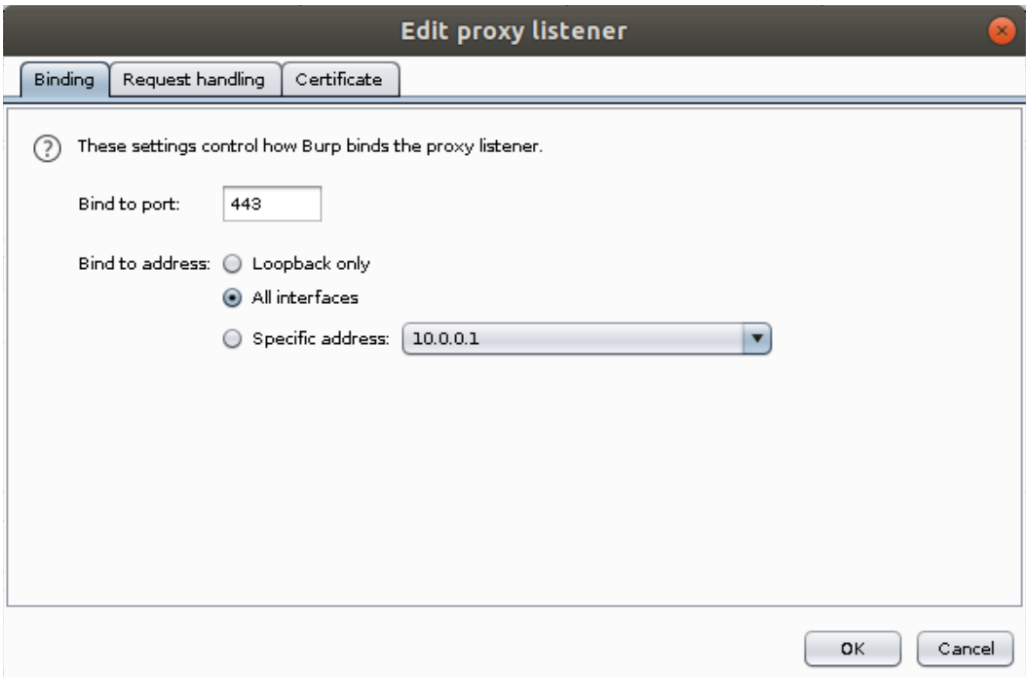
Po zapnutí vytvořím dočasný projekt. Poté se mi otevře okno projektu, zde uvidím veliké množství možných funkcí a nastavení. Některé jsou omezené

z důvodu neplacené verze programu a otevřou se až po zaplacení. Mě zajímá hlavně záložka **Proxy** a v ní okno **Options.**



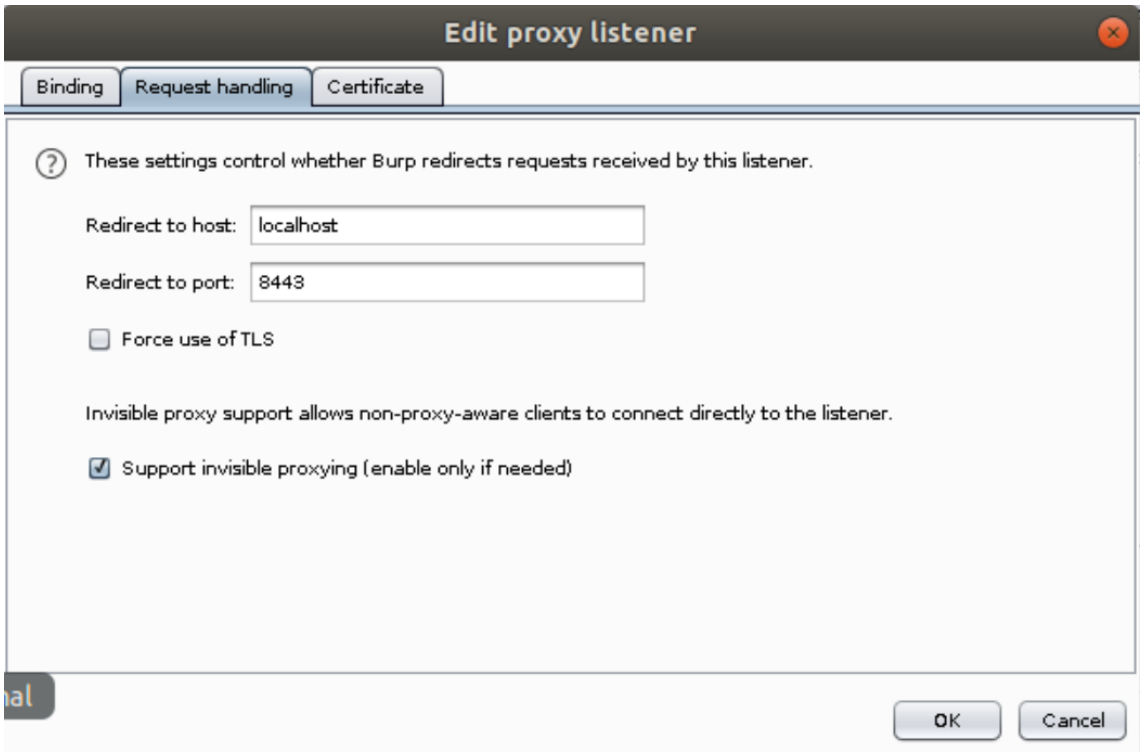
Obr. 7.8: Nastavení Proxy za pomoci programu BurpSuite

Zde přidám tlačítkem **Add** nové nastavení proxy. Otevře se mi okno viz (Obr. 7.8). Nastavím položku **Bind port** na **443**. Toto nastavení umožní naslouchat na portu 443. Mohl bych zde nastavit konkrétní adresu nebo síťový adaptér, jelikož mé virtuální stroje budou mít jen jedno připojení, tak můžu nastavit na **Bind address** na **All interfaces**. Díky tomu mám jistotu, že provoz z jakéhokoliv síťového adaptéru, který chce přistoupit k portu 443, bude přesměrován.



Obr. 7.9: Přesměrování komunikace z portu 443

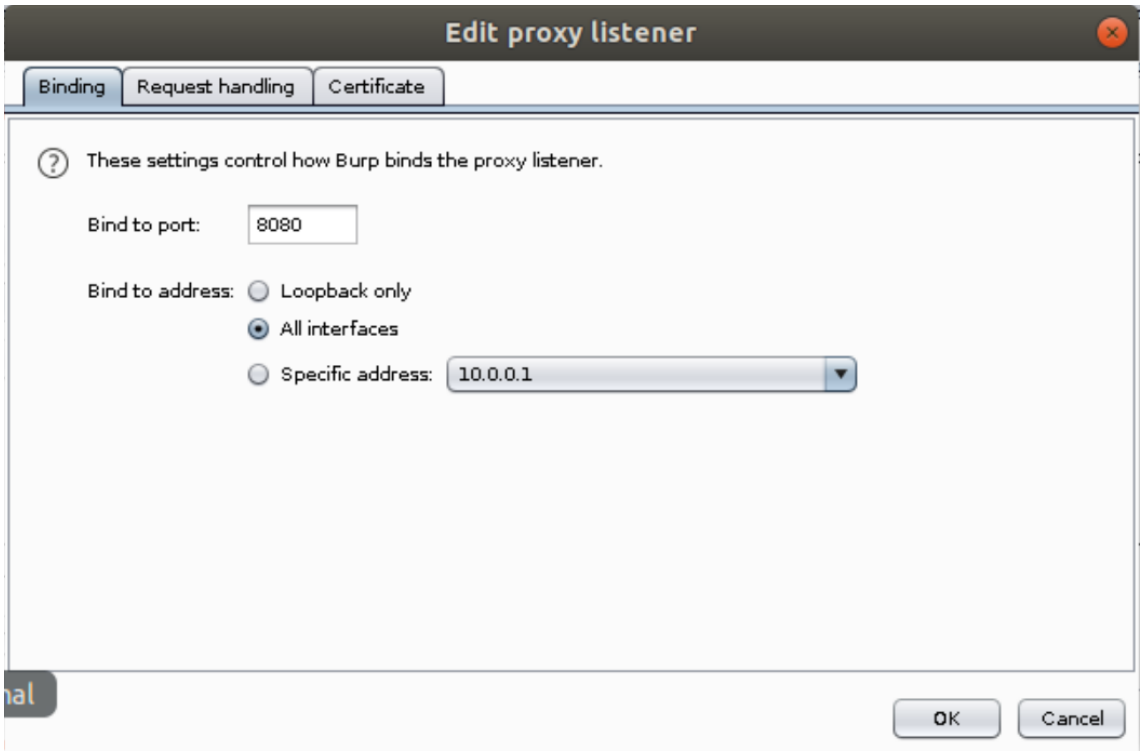
Samotné přesměrování nastavím v záložce **Requset handling**. Zde nastavím parametry **Redirect to host** na **localhost** a **Redirect to port** na mnou zvolený port **8443**, který jsem nastavoval i v programu INetSim. Dále zaškrtnu políčko u **Support invisible proxy**. Tím zamezím škodlivému kódu ve zjištění, že jde o proxy a že se děje něco podezřelého. Některý malware dokáže detekovat, když je přesměrován přes proxy a přestane se projevovat. Což by mohlo znamenat neúspěšnou analýzu.



Obr. 7.10: Přesměrování komunikace na port 8443

7.3.4 Importování Burp SSL certifikátu

Abych mohl přesměrovat a HTTPS komunikaci a tím jí sledovat, tak budu potřebovat přidat certifikát programu Burp. Nastavím nové proxy připojení na port **8080** a vyberu pro všechny **síťová rozhraní**. Až budu potřebovat stáhnout certifikát na infikované stroje, zadám do prohlížeče http://10.0.0.1:8080**.** Dále budu muset přidat Burp jako certifikátovou autoritu. To udělám tak, že si stáhnu certifikát přes prohlížeč. Poté musím přes nastavení prohlížeče importovat daný certifikát. Aby všechno fungovalo dobře, musím restartovat Burp a internetový prohlížeč.



Obr. 7.11: Nastavení proxy pro stažení CA certifikátu

Při práci s Linuxem, můžu použít některé příkazy, které mi mohou pomoct věci zjednodušit a urychlit. Díky programu openssl můžu změnit formát certifikátu z der na crt. Poté ho nakopíruji do složky s certifikáty, jiné pro různé distribuce Linuxu. Nakonec musím aktivovat nové nastavení certifikátů.

Výpis kódu 7.26: Importování certifikátů

```
$ openssl x509 -in ~/Downloads/cacert.der -inform DER -out burp.crt
#Pro Ubuntu:
$ sudo cp burp.crt /usr/local/share/ca-certificates/
#Pro Fedoru:
$ sudo cp burp.crt /usr/share/pki/ca-trust-source/
#Aktualizování certifikátů
$ sudo update-ca-certificates
```

8 Analýza škodlivého softwaru

8.1 Získání vzorků škodlivého softwaru

Získávání skutečných vzorků škodlivého softwaru může být určitou překážkou, jelikož neexistují volně přístupné databáze aktivních vzorků. Hodně kódu se dá najít na githubu, jak cvičné vzorky od lidí, které nejsou určené a v mnoha případech by byly nepraktické pro útoky. Dále jsou zde zdrojové kódy opravdových malwarů, některé jsou dokonce získané reverzním inženýrstvím souborů exe. Dále pak existují stránky jako je https://virusshare.com/, kde po registraci a vyplnění krátkého dotazníku obdržím email s přihlašovacími údaji.

Pro jednoznačnost a jasnost výsledků budu analyzovat vždycky pouze jeden vzorek škodlivého kódu. Rád bych předvedl analýzu více druhů škodlivého kódu. Budu používat nástroje u kterých jsem zde již mluvil, tudíž dále je už nebudu popisovat.

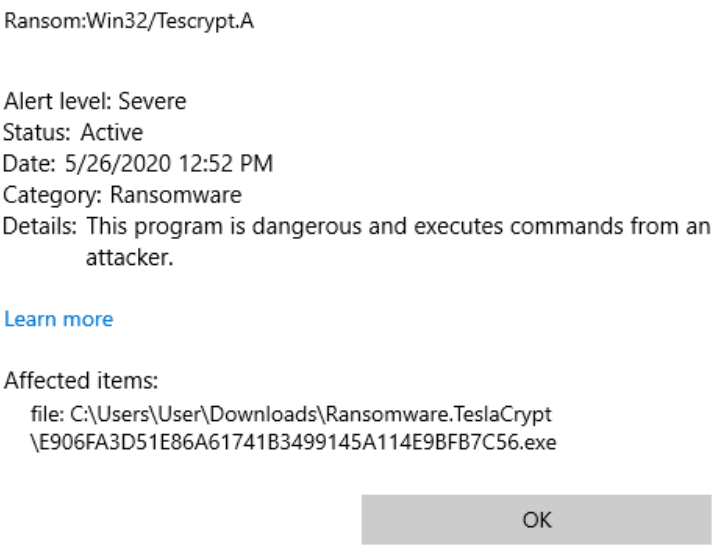
8.2 Ransomware

Jako ukázku analýzy ransomwaru, jsem vybral TeslaCrypt.

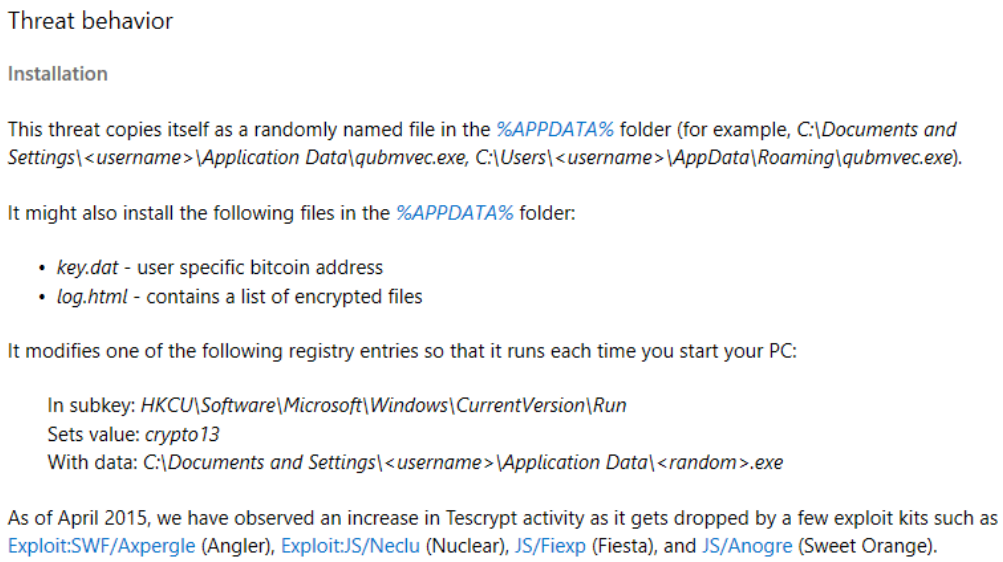
8.2.1 Statická analýza

Kontrola antivirovým softwarem

Jako první krok ve statické analýze provedu kontrolu daného vzorku antivirovými programy. Využiji zde antivirový systém Windows Defender a online nástroj VirusTotal. Windwos Defender mi hnedka nezobrazí nějakou hlubší analýzu nebo užitečné informace (Obr. 8.1). Nabízí však odkaz na stránky Microsoftu, které jsou plné užitečných informací, obzvláště pokud daný malware je již známý a jsou k němu přístupná data viz (Obr. 8.2).



Obr. 8.1: Detekce ransomwaru za pomoci Windows Defender



Obr. 8.2: Informace o Ransomware na stránkách Microsoftu

Jak (Obr. 8.1) tak i (Obr. 8.2) ukazuje, antivirový program správně určil mnou zvolený ransomware.

Nástroj Virus Total provede automatickou analýzu za použití různých antivirových nástrojů a jejich databází. Jak je vidět z (Obr. 8.3) i tento nástroj správně identifikoval ransomware, který jsem vybral pro tuto analýzu.

output.149787349.txt
myfile.exe
6d3d62a4cff19b4f2cc7ce9027c33be8_E906FA3D51E86A61741B3499145A114E9BFB7C56
6d3d62a4cff19b4f2cc7ce9027c33be8.None
afaba2400552c7032a5c4c6e6151df374d0e98dc67204066281e30e6699dbd18.bin
TeslaCrypt.exe
TeslaCrypt.bin
E906FA3D51E86A61741B3499145A114E9BFB7C56.exe
teslacrypt1.exe
Windows Update.exe.r

Obr. 8.3: Identifikace Ransomwaru za pomoci stránky VirusTotal.com

Testování použití šifrovacích nebo kompresních souborů

Tento test slouží k zjištění, zda mnou analyzovaný soubor s koncovkou exe, nepoužívá kompresní či šifrovací program k zamaskování či ztížení analýzy. Z (Obr. 8.4) je vidět, že tento soubor není zašifrován nebo komprimován.

Obr. 8.4: Detekování šifrování nebo komprese infikovaného souboru

Analýza knihoven použitých ransomwarem

Program Dependency Walker mi umožní rekurzivně sestavit strom závislostí v exe souboru. Díky němu uvidím, k jakým DLL knihovnám daný soubor přistupuje.

Obr. 8.5: Přehled načtených knihoven za pomoci programu Dependecy Walker

•KERNEL32.DLL – Nejdůležitější knihovna pro kernel operačního systému Windows. Většina funkci pracuje s touto knihovnou nějakým způsobem. Tato knihovna je požívaná pro práci s pamětí operace a systémovým přerušením.

Jedná se o nejčastěji používanou knihovnu, jejíž využití značí práci s pamětí, soubory či hardwarem.

ADVAPI32.DLL – Součást pokročilé API knihovny na podporu APIs včetně práce s registry a zebezpečovacími šifrovacími funkcemi.

SHELL32.DLL – Obsahuje funkce Windows Shell API, které jsou použité pro otevírání oken prohlížeče nebo souborů. Dále také může obsluhovat příkazový řádek.

SHLWAPI.DLL – Tato knihovna obsahuje funkce pro UNC,URL, přístup k registrům, nastavení barev.

NTDLL.DLL – Tato knihovna je volaná ostatními knihovnami například KERNEL32.DLL. Obsahuje NT systémové funkce, pro rozhraní Kernelu. Tato knihovna byla označená jako vážné bezpečnostní riziko. Pokud je tato knihovna použita, naznačuje to, že malware používá funkce, které nejsou programům standartně dostupné.

PSAPI.DLL – Knihovna sloužící k podpoře stavu procesů, využitá ve Windows Task Manageru.

USER32.DLL – Knihovna obsahující Windows API funkce zaměřené na uživatelské prostředí.

WININET.DLL – Knihovna obsahující funkce pro práci s internetem. Trojské koně se snaží maskovat jako tato knihovna a tento proces je brán jako bezpečnostní riziko.

8.2.2 Dynamická analýza

Analýza síťového provozu

Pro tento typ dynamické analýzy jsem vybral dva programy Wireshark a INetSim.

Program INetSim musím před samotnou analýzou zapnout a po jeho ukončení vytvoří soubor, který obsahuje veškerou komunikaci. Z tohoto logu jsem vybral pouze záznamy, které mají jistou spojitost s ransomwarem viz (Obr. 8.6). Ransomware provedl následující DNS vyhledání:

```
2020-05-25 21:12:25  DNS connection, type: A, class: IN, requested name: 7tno4hib47vlep5o.tor2web.blutmagie.de
2020-05-25 21:12:25  DNS connection, type: PTR, class: IN, requested name: 3.0.0.10.in-addr.arpa
2020-05-25 21:12:25  HTTPS connection, method: GET, URL: https://7tno4hib47vlep5o.tor2web.blutmagie.de/state.php?
U3ViamVjdD1QaW5nJmtleT1FNDI1MEFBNjgwNzExMDFEQzIwMjg0RjhGMDU1REVBMUEwNDkwRDdFMTE2NUUyNEFGMTk1MUJEQjk2QTBGMEY1JmFkZH
file name: /home/analytics/analysis/test-analysis/data/http/fakefiles/sample.html
2020-05-25 21:12:25  DNS connection, type: A, class: IN, requested name: 7tno4hib47vlep5o.tor2web.fi
2020-05-25 21:12:25  DNS connection, type: PTR, class: IN, requested name: 3.0.0.10.in-addr.arpa
2020-05-25 21:12:26  HTTPS connection, method: GET, URL: https://7tno4hib47vlep5o.tor2web.fi/state.php?
U3ViamVjdD1QaW5nJmtleT1FNDI1MEFBNjgwNzExMDFEQzIwMjg0RjhGMDU1REVBMUEwNDkwRDdFMTE2NUUyNEFGMTk1MUJEQjk2QTBGMEY1JmFkZH
file name: /home/analytics/analysis/test-analysis/data/http/fakefiles/sample.html
```

Obr. 8.6: Detekování podezřelé komunikace

HTTPS komunikaci s následující stránkami tor2web.org, tor2web.blutmagie.de a tor2web.fi. Tyto internetové služby umožňují přístup k Tor síti, bez toho, aniž by musel být nainstalován Tor prohlížeč. Dále proběhla snaha spojit se se serverem bitcoin.toshi.io, což je peněženka na kryptoměny. Díky tomu ransomware může poskytnout informace nutné k zaplacení výkupného a může dojít k ověření o provedení platby.

```
2020-05-29 08:37:30  HTTPS connection, method: GET, URL: https://34r6hq26q2h4jkzj.tor2web.fi/, file name: /home/a
2020-05-29 08:37:45  DNS connection, type: PTR, class: IN, requested name: 1.0.0.10.in-addr.arpa
2020-05-29 08:37:45  DNS connection, type: PTR, class: IN, requested name: 1.0.0.10.in-addr.arpa
2020-05-29 08:37:45  DNS connection, type: PTR, class: IN, requested name: 1.0.0.10.in-addr.arpa
2020-05-29 08:37:45  DNS connection, type: PTR, class: IN, requested name: 1.0.0.10.in-addr.arpa
2020-05-29 08:38:09  DNS connection, type: A, class: IN, requested name: bitcoin.toshi.io
2020-05-29 08:38:09  DNS connection, type: PTR, class: IN, requested name: 3.0.0.10.in-addr.arpa
2020-05-29 08:38:09  DNS connection, type: A, class: IN, requested name: www.inetsim.org
2020-05-29 08:38:09  DNS connection, type: AAAA, class: IN, requested name: www.inetsim.org
2020-05-29 08:38:09  HTTPS connection, method: GET, URL: https://bitcoin.toshi.io/api/v0/addresses/1QKB2NBCoqriGM
analysis/data/http/fakefiles/sample.html
2020-05-29 08:38:09  DNS connection, type: PTR, class: IN, requested name: 3.0.0.10.in-addr.arpa
2020-05-29 08:38:10  HTTPS connection, method: GET, URL: https://7tno4hib47vlep5o.tor2web.blutmagie.de/state.php?
U3ViamVjdD1QYXltZW50JnJlY292ZXJ5X2tleT0wM0FBRDAwQTg5NEY1QjNBQ0NBQTQ2MEY2RkQ5MjcwNzU3NEIyQ0Q3RTkzMENGRkEwQkU0MkM5N
file name: /home/analytics/analysis/test-analysis/data/http/fakefiles/sample.html
```

Obr. 8.7: Identifikování pokusu o komunikaci s bitcosho.io

Ransomware může také komunikovat se skrytými službami jako je 7tno4hib47vlep5o.onion, který slouží jako C&C server. Tato komunikace zde neproběhla.

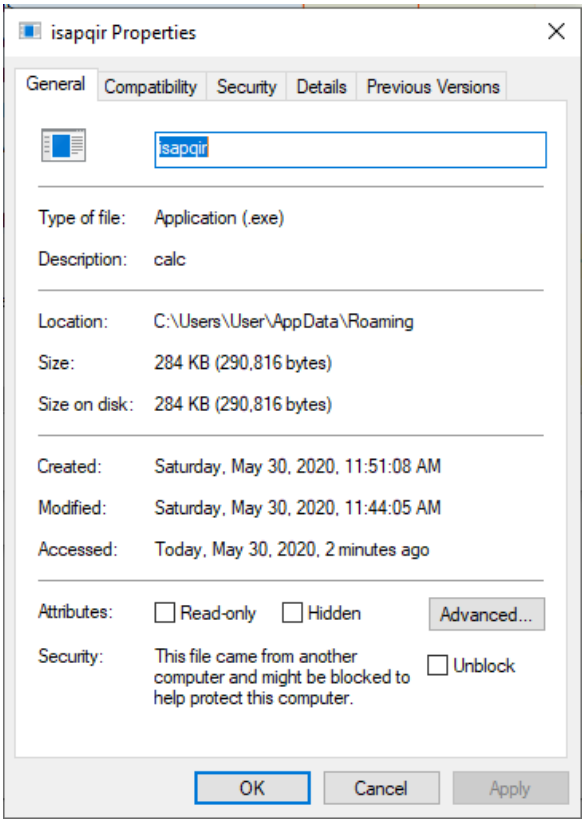
Program Wireshark mi poskytl podobné výsledky, jako program INetSim. Největší výhodou Wiresharku však zůstává množství informací, které se můžeme

o komunikaci dozvědět, avšak v tomto kroku nepotřebuji tak podrobné informace, stačí mi pouze záznamy o DNS a HTTPS komunikaci. Proto jsem zvolil INetSim jako hlavní program k zachycení síťového provozu.

Monitorování změn za pomoci aplikace Process Monitor

Monitorování změn během běhu malwaru mi může hodně ukázat o záměrech škodlivého kódu, jelikož mohu zaznamenat celý průběh útoku malwaru na daný systém.

Po spuštění infikovaného exe souboru, se zapne proces s názvem „calc“ viz (Obr. 8.8). Použitím správce úloh mi umožní zjisti umístění tohoto procesu a název ransomwaru, který je náhodně vygenerován.



Obr. 8.8: Zjištění přítomnosti podezřelého procesu s programem Správce úloh

Dojde k vytvoření dočasných souborů ve složce Roaming, které obsahují samotný malware. Dojde k vytvoření souboru key.dll, který obsahuje veřejný klíč pro RSA. Poté dojde k načtení potřebných systémových knihoven, jak je vidět na (Obr. 8.9). Podle načtených knihoven můžeme odhadovat záměry daného softwaru. Z (Obr. 8.10) vyplývá, že došlo k načtení kryptografických knihoven

cryptbase.dll a bcryptprimitives.dll. Dále malware používá k šifrování kryptografického poskytovatele **Microsoft Strong Cryptography Provider** typu PROV_RSA_FULL. TeslaCrypt rekurzivně prohledává každou složku a zašifruje její obsah. Všechny zašifrované soubory mají koncovku .ecc, takže přístup k nim má pouze ransomware. TeslaCrypt s největší pravděpodobností používá pro šifrování AES, z důvodů jeho rychlosti a praktičnosti.

Jako jednu z posledních věcí, ransomware vygeneruje soubory, které mají oznámit uživateli o tom že, zařízení bylo napadeno a vytvoří soubor HELP_TO_DECRYPT_YOUR_FILES.txt viz (Obr. 8.11). Nakonec je proces ukončen.

11:51...	isapqir.exe	6860	Load Image	C:\Users\User\AppData\Roaming\isapqir.exe	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\ntdll.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\System32\wow64.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\System32\wow64win.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\System32\wow64cpu.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\kernel32.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\KernelBase.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\apphelp.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\shlwapi.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\msvcrt.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\combase.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\ucrtbase.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\rpcrt4.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\sspicli.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\cryptbase.dll	SUCCESS
11:51...	isapqir.exe	6860	Load Image	C:\Windows\SysWOW64\bcryptprimitives.dll	SUCCESS

Obr. 8.9: Přehled načtených knihoven ransomwarem

11:51...	isapqir.exe	6860	CreateFile	C:\Windows\SysWOW64\bcrypt.dll
11:51...	isapqir.exe	6860	QuerySecurityFile	C:\Windows\SysWOW64\bcrypt.dll
11:51...	isapqir.exe	6860	QuerySecurityFile	C:\Windows\SysWOW64\bcrypt.dll
11:51...	isapqir.exe	6860	CloseFile	C:\Windows\SysWOW64\bcrypt.dll
11:51...	isapqir.exe	6860	QueryNameInfo...	C:\Windows\SysWOW64\bcrypt.dll
11:51...	isapqir.exe	6860	RegOpenKey	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001
11:51...	isapqir.exe	6860	RegSetInfoKey	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001\Name
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001\Name
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001\Name
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001\Name
11:51...	isapqir.exe	6860	RegCloseKey	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider Types\Type 001
11:51...	isapqir.exe	6860	RegOpenKey	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider
11:51...	isapqir.exe	6860	RegSetInfoKey	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider\Type
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider\Image Path
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider\Image Path
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider\Image Path
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Defaults\Provider\Microsoft Strong Cryptographic Provider\Image Path
11:51...	isapqir.exe	6860	RegOpenKey	HKLM\Software\Policies\Microsoft\Cryptography
11:51...	isapqir.exe	6860	RegSetInfoKey	HKLM\SOFTWARE\Policies\Microsoft\Cryptography
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\Policies\Microsoft\Cryptography\PrivKeyCacheMaxItems
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\Policies\Microsoft\Cryptography\PrivKeyCachePurgeIntervalSeconds
11:51...	isapqir.exe	6860	RegQueryValue	HKLM\SOFTWARE\Policies\Microsoft\Cryptography\PrivateKeyLifetimeSeconds

Obr. 8.10: Načtení kryptografických knihoven

11:55...	isapqir.exe	6860	ReadFile	C:\Windows\SysWOW64\windows.storage.dll	SUCCESS
11:55...	isapqir.exe	6860	WriteFile	C:\Users\User\Desktop\CryptoLocker.lnk	SUCCESS
11:55...	isapqir.exe	6860	CloseFile	C:\Users\User\Desktop\CryptoLocker.lnk	SUCCESS
11:55...	isapqir.exe	6860	RegQueryKey	HKCU	SUCCESS
11:55...	isapqir.exe	6860	RegQueryKey	HKCU	SUCCESS
11:55...	isapqir.exe	6860	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\App Paths\isapqir.exe	NAME NOT FOUND
11:55...	isapqir.exe	6860	RegQueryKey	HKLM	SUCCESS
11:55...	isapqir.exe	6860	RegQueryKey	HKLM	SUCCESS
11:55...	isapqir.exe	6860	RegOpenKey	HKLM\Software\WOW6432Node\Microsoft\Windows\CurrentVersion\App Paths\isapqir.exe	REPARSE
11:55...	isapqir.exe	6860	RegOpenKey	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\isapqir.exe	NAME NOT FOUND
11:55...	isapqir.exe	6860	Thread Create		SUCCESS
11:55...	isapqir.exe	6860	CreateFile	C:\Users\User\Desktop\HELP_TO_DECRYPT_YOUR_FILES.txt	SUCCESS
11:55...	isapqir.exe	6860	QueryStandardI...	C:\Users\User\Desktop\HELP_TO_DECRYPT_YOUR_FILES.txt	SUCCESS
11:55...	isapqir.exe	6860	WriteFile	C:\Users\User\Desktop\HELP_TO_DECRYPT_YOUR_FILES.txt	SUCCESS
11:55...	isapqir.exe	6860	WriteFile	C:\Users\User\Desktop\HELP_TO_DECRYPT_YOUR_FILES.txt	SUCCESS
11:55...	isapqir.exe	6860	CloseFile	C:\Users\User\Desktop\HELP_TO_DECRYPT_YOUR_FILES.txt	SUCCESS
11:55...	isapqir.exe	6860	ReadFile	C:\Windows\SysWOW64\wininet.dll	SUCCESS
11:55...	isapqir.exe	6860	RegQueryValue	HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\NameSpace_Catalog5\Serial_Access_Num	SUCCESS
11:55...	isapqir.exe	6860	RegQueryKey	HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\NameSpace_Catalog5	SUCCESS
11:55...	isapqir.exe	6860	RegOpenKey	HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\NameSpace_Catalog5\00000016	NAME NOT FOUND
11:55...	isapqir.exe	6860	RegQueryValue	HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\Protocol_Catalog9\Serial_Access_Num	SUCCESS
11:55...	isapqir.exe	6860	RegQueryKey	HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\Protocol_Catalog9	SUCCESS
11:55...	isapqir.exe	6860	RegOpenKey	HKLM\System\CurrentControlSet\Services\WinSock2\Parameters\Protocol_Catalog9\0000000C	NAME NOT FOUND
11:55...	isapqir.exe	6860	Thread Create		SUCCESS
11:55...	isapqir.exe	6860	Thread Exit		SUCCESS

Obr. 8.11: Vytvoření textového souboru s instrukcemi a ukončení programu

Porovnání změn v registrech Aplikací Regshot

Tato analýza mi poskytne podobné informace, jako program Process Monitor. Má své výhody, ale i nevýhody. Abych dostal co nejoptimalizovanější výsledky, tak je dobré první „originální“ obraz registrů provést co v nejkratší době před infikováním testovaného stroje. Poté co malware udělá veškerou škodu, pořídím

druhý obraz. Tyto obrazy jsou buď v HTML nebo txt formátu. Program je poté porovná a vypíše veškeré změny oproti „originálnímu“ obrazu.

Hlavní nevýhodou může být ohromné množství zápisů, jako třeba na (Obr. 8.12) , kde jen bylo smazáno 61153 klíčů v registrech. To je důsledkem mnou analyzovaného ransomwaru, u kterého je pravděpodobné že dojde k mnoha zápisům. Pokud se však budu potýkat s něčím jako je trojský kůň nebo nějaký spyware, tak změn v registrech nebude tolik a umožní mi to jednodušeji odhalit provedené změny, jaké můžeme vidět na (Obr. 8.13).

Obr. 8.12: Případné zahlcení záznamů programu Regshot

Obr. 8.13: Změna pozadí a vytvoření instrukcí pro postiženého

8.2.3 Shrnutí

Pokud máme na počítači nějaká osobní data, tak škody napáchané ransomwarem jsou veliké. Jakmile jsou data jednou zašifrovaná, tak už nikdy nemusí být rozšifrovaná. Většinou jen u starších ransomwaru byla prolomena šifra, aby oběti mohli bezplatně obnovit data. Tím, že ransomware se chová agresivně, nechává za sebou spoustu stop, které jsou jednoduché na detekování. Tím je jeho analýza jednodušší než více nenápadné malwary. Povedlo se mi úspěšně detekovat zvolený ransomware, odhalit snahu o komunikaci a poukázal jsem na důležité části průběhu ransomwaru.

8.3 Trojský kůň

Jako druhou ukázku jsem vybral druh stahovacího trojského koně. Jeho název je .GLUPTEBA.TA. Jde o poměrně nový malware, je sofistikovaný a snaží komplikovat jeho možnou analýzu.

8.3.1 Statická analýza

Kontrola antivirovým softwarem

Windows Defender byl schopen odhalit nebezpečný soubor a určit malware, který je v něm obsažen. Bohužel databáze Microsoftu neobsahuje žádné užitečné informace, které bych později mohl použít. V tomto případě mi pomůže VirusTotal, kde mohu nají spoustu užitečných informací. Třeba o možném síťovém provozu nebo o možných změnách v registrech.

Obr. 8.14: Identifikace trojského koně programem Windows defender

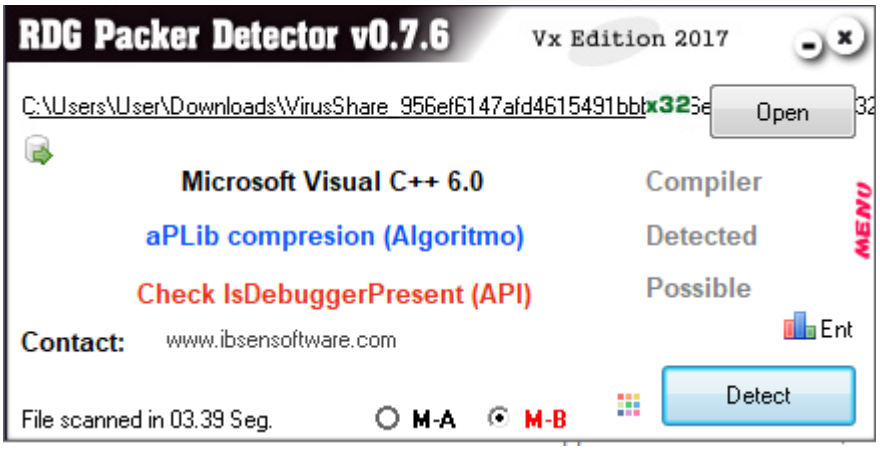
Basic Properties

MD5	956ef6147afd4615491bbb25a6ee08c2
SHA-1	3fe9172738c00aef54d2aec35ece0c388ac8d213
SHA-256	3586d83281a526ee8b46710bafd593cf8671d2f6c7598237086dee41f1cf437b
Vhash	056056655d65556055z6004b7z1dz8fz
Authentihash	56a7dc49f4c2c7b096351b222052742c5e5e83dc37656e8c74e02ad331ecbf74
Imphash	23e9c5e44202fc0032ec48905f47bea5
SSDEEP	98304:KZRmedR3DtSjJFqfsAuXRt6JMVtjFvCIRGT5+aNN4fzB6WID:BedR3ojJFwC6GtjF6BT5+a2N
File type	Win32 EXE
Magic	PE32 executable for MS Windows (GUI) Intel 80386 32-bit
File size	4.90 MB (5137920 bytes)

Obr. 8.15: Identifikace trojského koně stránkou VirusTotal

Testování použití šifrovacích nebo kompresních souborů

Podle výsledků tohoto testu soubor exe, který jsem použil pro infikování analyzovaného stroje, je komprimován pomocí aPLib. Existují dekompresní programy, které jsou volně dostupné. Tato kompresní knihovna se používá pro zdrojové kódy psané v jazyce C, což je vidět i z řádku „Compiler.“



Obr. 8.16: Detekování komprese infikovaného souboru algoritmem aPLib

Analýza knihoven použitých trojským koněm

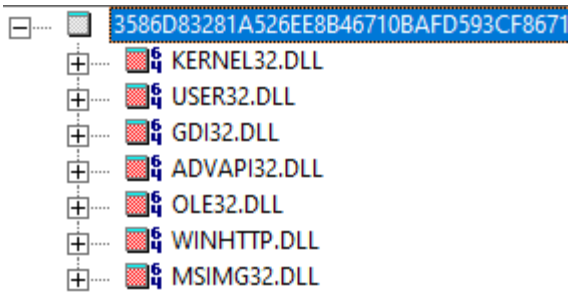
Z knihoven používaných malwarem jsou asi nejzajímavější knihovny winnet.dll a advapi32.dll. Winnet.dll ukazuje, na snahu malwaru komunikace přes http protokol.

ADVAPI32.DLL – Součást pokročilé API knihovny na podporu APIs včetně práce s registry a zebezpečovacími šifrovacími funkcemi.

USER32.DLL – Knihovna obsahuje Windows API funkce zaměřené na uživatelské prostředí.

WININET.DLL – Knihovna obsahuje funkce pro práci s internetem. Trojské koně se snaží maskovat jako tato knihovna a tento proces je brán jako bezpečnostní riziko.

WINHTTP.DLL – Knihovna pro Windows HTTP Servis, nesystémová knihovna. Je instalovaná se softwarem třetích stran.



Obr. 8.17: Zjištění knihoven používaných trojským koněm

8.3.2 Dynamická analýza

Analýza síťového provozu

Pro tuto analýzu jsem zvolil program Wireshark. Jak je ale na první pohled možné vidět z (Obr. 8.18), malware provedl jen jediný pokus o komunikaci s internetovou stránkou venoxcontrol.com.

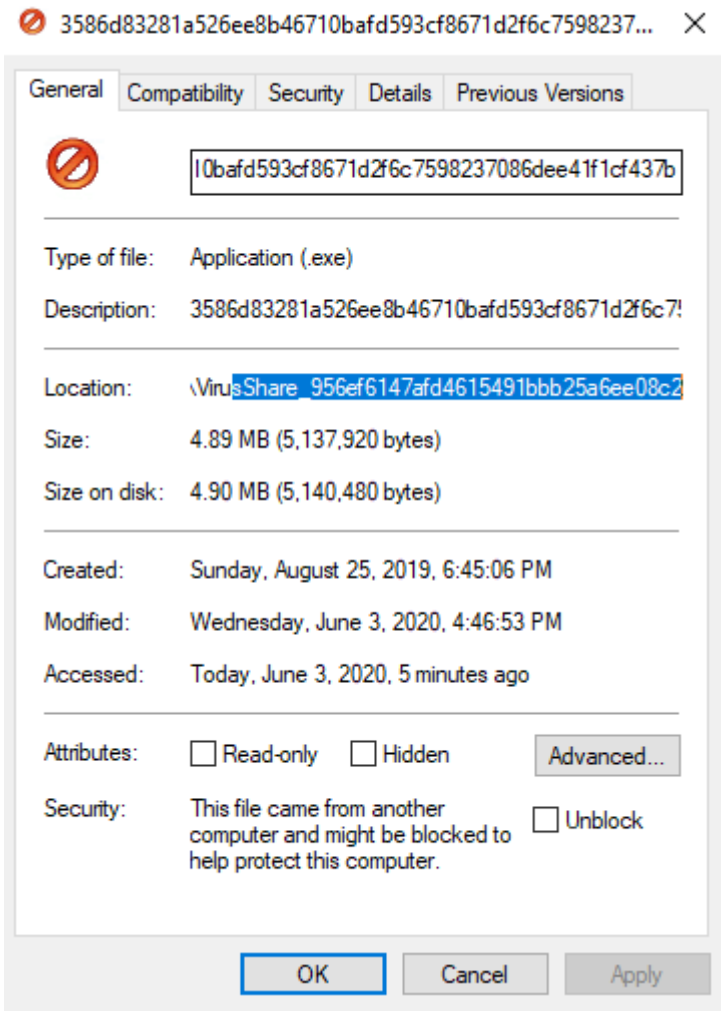
Ta slouží jako kontrolní uzel pro tento typ trojského koně. Jelikož malware nedostal adekvátní odpověď od tohoto serveru, tak mohla být naplněna podmínka a již se nesnaží komunikaci opakovat. To by mělo znepříjemnit snahu analyzovat chování tohoto malwaru.

```
10.0.0… 10.0.0.1 DNS     76 Standard query 0xb5da A venoxcontrol.com
10.0.0… 10.0.0.3 DNS     92 Standard query response 0xb5da A venoxcontrol.com A 10.0.0.1
10.0.0… 10.0.0.1 DNS    132 Standard query 0x35cb PTR 1.6.7.2.b.6.2.4.d.5.2.d.0.8.7.a.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa
10.0.0… 10.0.0.3 DNS    161 Standard query response 0x35cb PTR 1.6.7.2.b.6.2.4.d.5.2.d.0.8.7.a.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa PTR www.
```

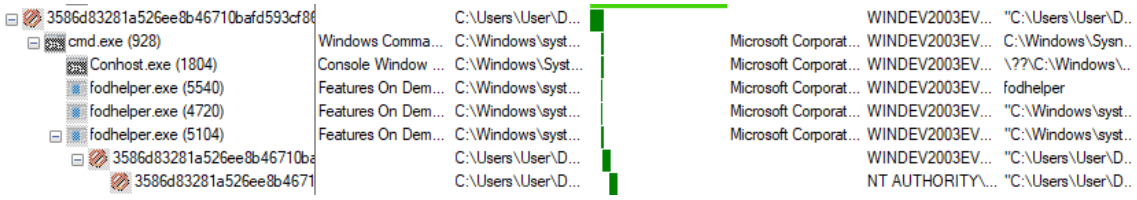
Obr. 8.18: Úspěšné odhalení pokusu komunikovat s C&C serverem

Monitorování změn za pomoci aplikace Process Monitor a Reg Shot

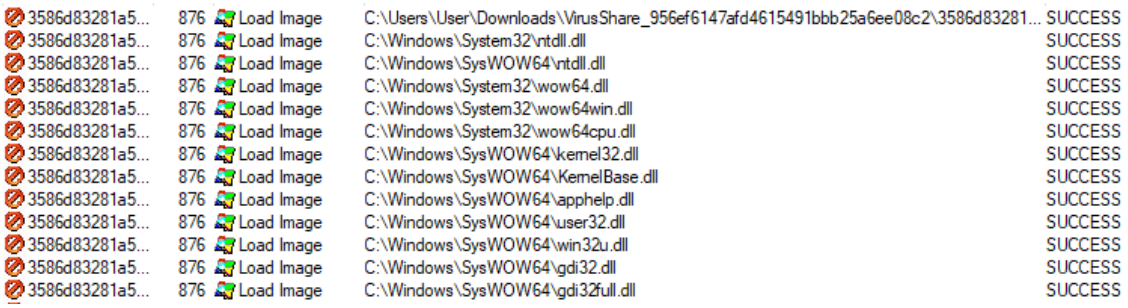
Po zapnutí podezřelého exe souboru, se nastartuje proces, který má stejné jméno jako exe soubor. Pro mě je to hash SHA256 tohoto malwaru. Jak je vidět z (Obr. 8.20), program spouští řadu podprocesů. Ty mohu monitorovat stejným způsobem jako hlavní proces. Díky programu Process Monitor je vidět, jak jednotlivé podprocesy běželi v čase, kdy se zpustily, kde běžely a pod jakými uživatelskými právy běží. Každý podproces si načte potřebné knihovny (Obr. 8.21).



Obr. 8.19: Zjištění běhu podezřelého souboru programem Správce úloh

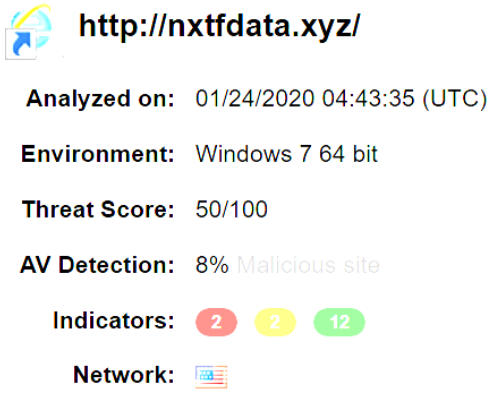


Obr. 8.20: Detekování stromu procesů trojského koně programem Process Monitor

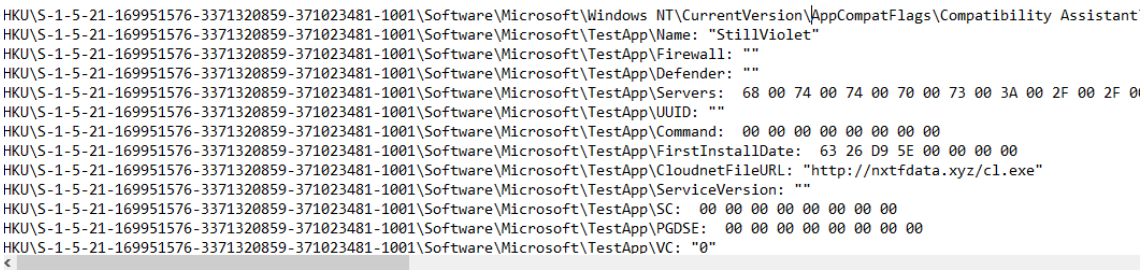


Obr. 8.21: Záznam načtených knihoven hlavním procesem trojského koně

Z následujícího zápisu do registrů viz (Obr. 8.23) , malware pracuje s registry umožňující programu spuštění v módu kompatibility. Jelikož zápis toho řádku byl příliš dlouhý, tak jsem musel odříznout za slovem „Assistant“ cestu k souboru s obsahující malware, aby se vešel na stránku. Malware je navržen pro Windows 7. Dále je ze zápisu patrné, že došlo k vytvoření aplikace se jménem StillViolet. Zajímavým zápis je na řádku, který obsahuje „CloudnetFileURL:“ Odkaz http://nxtfdata.xyz/cl.exe je považována za nebezpečný soubor (Obr. 8.22).



Obr. 8.22: Bezpečnostní analýza podezřelé stránky



Obr. 8.23: Důkaz o použití režimu kompatibility trojským koněm

8.3.3 Shrnutí

Největší komplikací u analýzy trojského koně, co by měl sloužit k stažení dalších malwarů, je přístup k internetu. V mém analytickém prostředí se ho snažím simulovat programem INetSim, problém nastává v tom, když se malware snaží komunikovat s řídícím uzlem neboli C&C serverem. Malware potom nedostává pokyny, odkazy a další informace. Tím přestane plnit jeho hlavní roli. Jelikož tato komunikace neprobíhá, tak nemohu přesně určit další potenciální zdroje vysílání.

Tento malware jsem vybral záměrně abych poukázal na možné problémy s malwarem, co se snaží zůstat nedetekován. I přesto, že nejsem schopen s přesností určit možné další následky malwaru, tak moje prostředí bylo schopno určit potencionální hrozbu. Pokud bych zde chtěl zlepšit moje analytické prostředí, tak by bylo nejvhodnější zajistit bezpečný přístup na internet. To může být komplikované, jelikož některé viry napadají i přístupové body.

8.4 Ochrana proti škodlivému softwaru

Mohl bych zde mluvit o obecných zásadách, jak člověk nemá otevírat podezřelé odkazy, klikat na spamové emaily, popřípadě spouštění neznámých programů stažených třeba přes torrenty. Předpokládám však, že čtenář této práce je s těmito principy obeznámen.

V dnešní době představují největší hrozbu takzvané „zero day vulnerability“ neboli chyby v programech, které se nacházejí hnedka při vydání. Není proti nim ochrana, jelikož vývojáři na tuto chybu ještě nenarazili a nemohli tak zakomponovat identifikační prvky do antivirových řešení.

Nejlepší ochrana před ransomwarem je nenechat nakazit daný systém. Pokud už se tak stalo, jedinou věc, co člověk může udělat je, snažit se minimalizovat napáchané škody, prostřednictvím preventivních záloh systému a osobních dat. Operační systém Windows 10 nabízí jako možnost obrany před ransomwarem zálohování dat přes One Drive. Další antivirové programy nabízejí podobné funkce.

Existuje zde ještě jedna možnost, jak se bránit proti ransomwaru. V mnou navrženém prostředí by šel detekovat a rozpoznat takzvaný „kill switch“, což je

internetová doména, se kterou ransomware komunikuje, pokud je neaktivní tak jí ransomware ignoruje. Pokud by se pokusil komunikovat a zjistil že URL adresa je aktivní, tak se deaktivoval, jako to bylo u ransomwaru WannaCry.

Pokud jde o obranu před trojskými koni, tak samozřejmě platí základy. Dobrý antivirový program by měl odhalit většinu snah o infikaci zařízení. Je tu ještě možnost aktivního přístupu a přidávat podezřelé IP a URL adresy na „blacklist“ síťových prvků jako jsou wifi routery switche apt. Tím se zamezí komunikace s řídícími uzly malwaru, takže nebudou moct stáhnout další nežádoucí programy na zařízení. Aktivní filtrace na přístupových bodech je však výpočetně náročná a je potřeba nákladnější přístupový bod. Další zajímavou možností, která ovšem není pro každého, je použití vlastního filtrovacího zařízení, které je zapojené mezi hlavním přístupovým bodem a internetem. Je možné pronajmout si malý server a za pomoci VPN si přešněrovat komunikaci tam, provést filtraci a poté zase zpět. Další možnost je použití malého počítače jako je Raspberry Pi a poté použít jednoduchý DNS server Pi-hole. Nevýhodou těchto řešení je jednak náročnost jak z technického hlediska, tak i z hlediska udržování aktuálnosti filtrů. Další problém může nastat, pokud se s setkáme s novým malwarem, který ještě není zanalyzovaní a může požívat ještě neznáme IP nebo URL adresy.

Závěr

Na závěr bych rád shrnul poznatky obsažené v této práci, tedy obecné seznámení čtenáře s problematikou nebezpečí škodlivého kódu. Rizika, které představuje moderní spyware a trojské koně, uvedl jsem, jak rychle dochází k rozšiřování škodlivého kódu a jakou škodu dokáže napáchat. Toto demonstruji na datech ze studií, ve formě tabulek a grafů.

Podstatnou část mé práce představuje, návod postupu při vytváření vlastního analytického prostředí jak pro platformu Windows 10 tak i pro Fedoru. Věnuji se také jako vytváření virtuálního stroje za pomoci grafického prostředí, tak i za výhradního použití konzole a příkazů pracujících v prostředí bash a powershell. Použití jednotlivých příkazů odůvodňuji, a vysvětluji proč jsem je použil. V jedné kapitole popisuji, jaké analytické nástroje použiji a jak si je stáhnout a nainstalovat.

V druhé části práce se věnuji vlastní analýze mnou vybraných škodlivých kódů. Ukazuji postup analýzy a důležité informace o průběhu malwaru, které nasvědčují podezřelé chování malwaru. Analyzuji síťovou komunikaci, z toho vyvozuji závěr, jestli se jedná o potencionálně nebezpečnou komunikaci. Na závěr mluvím o možnostech obrany před analyzovanými viry. Navrhuji zde i komplikovanější řešení filtrování příchozí komunikace za pomoci přesměrování a použití vlastního DNS serveru.

Hlavním cílem mojí práce bylo ukázat, jak připravit bezpečné prostředí pro analýzu škodlivého softwaru. Jak jsem ukázal na vzorcích malwaru, tak můj sandbox funguje. Dokáže identifikovat možnou hrozbu, zaznamená síťovou komunikaci a monitorovat změny provedené malwarem.

Literatura

[1] ŠTĚPÁNKOVÁ, KRISTINA, 2019, *Dark Avenger: Chápání počítačového viru a motivace k jeho tvorbě optikou hackera*. Bc. Masarykova univerzita.

[2] PERRIN, CHAD, 2006, Hacker vs. cracker. *techrepublic* [online]. 2006. [Accessed 20 December 2019]. Available from: https://www.techrepublic.com/blog/it-security/hacker-vs-cracker/

[3] What is the Difference Between Black, White and Grey Hat Hackers?, 2019. *Us.norton.com* [online],

[4] SPOFFORD, EUGENE H, 1988, *The Internet worm program*. West Lafayette, IN : Purdue University, Dept. of Computer Sciences.

[5] FEDERAL TRADE COMMISSION, 2005, *Monitoring Software on Your PC: Spyware, Adware, and Other Software*. Federal Trade Commission.

[6] JOHANOVSKÝ, TOMÁŠ, 2018, *Kriminologické a trestněprávní aspekty fenoménu ransomware*. Mgr. UNIVERZITA KARLOVA.

[7] JIROVSKÝ, VÁCLAV, 2007, *Kybernetická kriminalita: nejen o hackingu, crackingu,virech a trojských koních bez tajemství*.. Praha : Grada.

[8] KOPŘIVA, JAN, 2019, *Historie a vývoj malware*. Bc. Unicorn College.

[9] BLUNDEN., BILL, 2009, *The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System*. 1. Jones and Bartlett Learning.

[10] TRUTMAN, MICHAL, 2019, *ROOTKIT PRO MS WINDOWS*. Vysoké Učení Technické v Brně.

[11] PONEMON INSTITUTE LLC, 2017, *COST OF CYBER CRIME STUDY: INSIGHTS ON THE SECURITY INVESTMENTS THAT MAKE A DIFFERENCE* [online]. [Accessed 20 December 2019]. Available from: https://www.accenture.com/t20170926t072837z__w__/us-en/_acnmedia/pdf-61/accenture-2017-costcybercrimestudy.pdf

[12] SIKORSKI, MICHAEL and HONIG, ANDREW, 2012, *Practical malware analysis*. San Fransisco, CA. : No Starch Press Inc.

[13] BALLES, CHRIS and SHARFUDDIN, ATEEQ, 2019, *Breaking Imphash* [online]. [Accessed 20 December 2019]. Available from: https://arxiv.org/ftp/arxiv/papers/1909/1909.07630.pdf

[14] VirusTotal, 2019. *Virustotal.com* [online],

[15] BRONIEK, DAVID, 2019, *Analýza malware*. Bc. VŠB – Technická univerzita Ostrava.

[16] LIGH, MICHAEL, RICHARD, MATT and ADAIR, STEVEN, 2010, *Malware Analyst?s Cookbook and DVD: Tools and Techniques for Fighting Malic*. John Wiley & Sons.

[17] Process Monitor – Windows Sysinternals, 2019. *Docs.microsoft.com* [online],

[18] How to Use Regshot To Monitor Your Registry, 2019. *How-To Geek* [online],

[19] WATSON, JON, 2019, What is sandboxing and how to sandbox a program. *comparitech* [online]. 2019. [Accessed 21 December 2019]. Available from: https://www.comparitech.com/blog/information-security/what-is-sandboxing/

[20] CHEBYSHEV, VICTOR and SINITSYN, FEDOR, 2019, IT threat evolution Q1 2019. Statistics. *Securelist.com* [online]. 2019. [Accessed 21 December 2019]. Available from: https://securelist.com/it-threat-evolution-q1-2019-statistics/90916/

[21] KUJAWA, ADAM and ZAMORA, WENDY, 2019, [online]. Malwarebytes. [Accessed 21 December 2019]. Available from: https://resources.malwarebytes.com/files/2019/01/Malwarebytes-Labs-2019-State-of-Malware-Report-2.pdf

[22] Chapter 20. Managing Guest Virtual Machines with virsh Red Hat Enterprise Linux 7 | Red Hat Customer Portal, 2019. *Red Hat Customer Portal* [online],

[23] Invoke-WebRequest (Microsoft.PowerShell.Utility), 2019. *Docs.microsoft.com* [online],

[24] MOSER, Andreas; KRUEGEL, Christopher; KIRDA, Engin. Limits of static analysis for malware detection. In: Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007). IEEE, 2007. p. 421-430.

[25] *Internet Users' Glossary* [online]. Austin, 1993 [cit. 2020-06-08]. Dostupné z: https://tools.ietf.org/html/rfc1392

[26]

Seznam symbolů, veličin a zkratek

KVM	Kernel-based Virtual Machine
GUI	Grafické uživatelské rozhraní - Graphical User Interface
UEFI	jednotné rozšiřitelné firmwarové rozhraní - Unified Extensible Firmware Interface
RPM	Red Hat Package Manager
RAM	paměť s přímým přístupem - Random-Access-Memory
VHD	virtuální pevný disk - Virtual Hard Disk
API	Application Programming Interface
URL	jednotná adresa zdroje - Uniform Resource Locator
CEH	Certified Ethical Hacker